SCORM

Best Practices Guide for Content Developers

1st Edition

2003-02-28

UPDATED!

This guide includes updates to Section 7; dated 2004-04-05.

Carnegie Mellon

Learning Systems Architecture Lab

Credits and Redistribution Guidelines

The *SCORM Best Practices Guide for Content Developers* is a publication of the Learning Systems Architecture Lab at Carnegie Mellon University.

This guide was sponsored by the Technical Support Working Group of the Combating Terrorism Technology Support Office and was created as a service to the e-learning community in an attempt to further the adoption of the Sharable Content Object Reference Model (SCORM).

For hyperlinks to this document, the appropriate URL is:

http://www.lsal.cmu.edu/lsal/expertise/projects/developersguide/

If you are redistributing or hyper-linking to this document, please notify us so that we may keep you informed of the latest revisions to the guide. For more information, please contact us.

Learning Systems Architecture Lab
Carnegie Mellon University
700 Technology Drive
Pittsburgh, Pennsylvania, USA 15219

Voice: +1.412.268.5322
Fax: +1.412.268.4772
Email: lsal@cmu.edu
Web: http://www.lsal.cmu.edu/

This Page Intentionally Left Blank

Preface 1

Using This Guide 3

1 Introducing SCORM 5

2 Defining Sharable Content Objects (SCOs) 9
2.1 Assets 9
2.2 Sharable Content Objects (SCOs) 9
2.3 SCOs and Reusability 11

3 Planning for SCORM 13
3.1 Building Your Development Team 13
3.2 Naming and Storing Your Files 14
3.3 Identifying Your Materials 15
3.4 Locating Your Instructional Materials 16
3.5 Using Tools to Design Your SCOs 21

4 Identifying and Designing Sharable Content Objects (SCOs) 23
4.1 Ensuring the Instructional Integrity of SCORM Content 23
4.2 Moving from a Traditional Course Structure to SCORM 24
4.3 Designing SCOs from Existing Instructional Materials 25
4.4 Designing SCOs for New Instructional Materials 28

5 Structuring Tests in SCORM 31

6 Navigating in SCORM 33
6.1 The User Interface 33
6.2 SCORM Navigation 33

7 Sequencing Your Content 35
7.1 Understanding Sequencing Terminology 36
7.2 Beginning with Simple Sequencing Templates 37
7.2.1 Template 1: Single SCO 40
7.2.2 Template 2: SCO with Assets 41
7.2.3 Template 3: The Black Box 42
7.2.4 Template 4: Multiple SCOs with Assets 43
7.2.5 Template 5: Remediating Using Objectives 45
7.2.6 Template 6: Pre- and Post-Test Sequencing 47

7.2.7 Template 7: Pre- and Post-Test Sequencing (2) 48
7.2.8 Template 8: Remediating Using Objectives (2) 50
7.2.9 Template 9: Basic Three-Way Branching 51
7.2.10 Template 10: Pre- and Post-Test Sequencing with New Content for Remediation....52
7.3 Building Instructional Models Using the Templates 54
7.3.1 Model 1: Remediating Multiple Aggregations 55
7.3.2 Model 2: Mastery Testing Multiple Aggregations 58
7.3.3 Model 3: Pre- and Post-Test Sequencing with Aggregations 60
7.3.4 Model 4: Traditional CBT Branching with Multiple Decisions 62
7.4 Creating Display Names 64
7.5 Communicating Aggregation Information to the Programmer....65

8 Packaging Your Content 67

9 Appendix 69
9.1 LSAL Template for SCO Design Specifications....70
9.2 LSAL Template for Aggregation Design Specifications....71

10 Glossary 73

11 References.... 77

12 Summary of Significant Changes in 1st Edition 79

Preface

This is a best practices guide created specifically for content developers and instructional designers tasked with creating new instructional materials that comply with the Sharable Content Object Reference Model (SCORM) or converting existing instructional materials into SCORM-compliant materials. However, writers, programmers, and subject matter experts will also find the guide a useful companion to the SCORM documents.

> **Compliance vs. Conformance**
> The words compliance and conformance, which are often used interchangeably in regular English to convey "accordance", have distinct meanings when used in the context of the adherence of something to a specification or standard. While there is a formal certification program to test for strict conformance of any product to the SCORM guidelines, it is not assumed that the products derived from the processes or procedures in the guide will automatically pass the conformance tests. Thus, this document uses "SCORM-compliant" to refer to the development of materials that comply with the written SCORM guidelines but that are not certified as "SCORM-conformant."

The guide provides a systematic process for using SCORM and tips to make your SCORM implementation easier. It is not intended to replace the SCORM documents, nor is it intended to be all-inclusive. The tips and techniques explained here will facilitate your entry into SCORM-compliant training, but through your own implementation, you will continue to learn more about the ways you can efficiently create effective SCORM-based instructional materials.

While SCORM claims to be pedagogically neutral, this guide focuses specifically on a single-user, self-paced e-learning pedagogy; that is, one learner interacting with the instruction. The guide's primary focus is for the training community; however, the SCORM definitions and strategies presented in the guide can be easily transferred to the educational community for a wide range of learners (including K-12 and higher education). The guide can also be applied to developing materials for distance education, computer-assisted instruction (CAI), and some forms of classroom instruction.

There has been considerable debate about what a sharable content object (SCO) is, how big it should be, and what it should contain (objectives, assessments, simulations, etc.). There are no concrete answers to any of these questions. In many cases, you'll need to adapt your SCOs to fit your specific needs. This guide attempts to provide a somewhat flexible definition of a SCO while providing some best practices to help you create the types of SCOs that will best meet your organization's specific needs. It also provides sequencing templates and models to assist you in creating an effective instructional design that complies with SCORM 1.3.

This Page Intentionally Left Blank

Using This Guide

The *SCORM Best Practices Guide for Content Developers* is organized as eight sections plus one appendix, as follows:

Section 1 **Introducing SCORM**
This section provides background information on SCORM and discusses the four SCORM "ilities".

Section 2 **Defining Sharable Content Objects (SCOs)**
This section defines Sharable Content Objects as well as other key terms that will be used throughout the guide.

Section 3 **Planning for SCORM**
This section addresses some of the people, processes, and tools you should have in place before attempting to create SCORM-compliant content.

Section 4 **Identifying and Designing Sharable Content Objects (SCOs)**
This section explains the design processes for converting existing content to SCORM and for creating new content that is SCORM-compliant.

Section 5 **Structuring Tests in SCORM**
This section discusses how you can assimilate your organization's testing requirements into SCORM.

Section 6 **Navigating in SCORM**
This section addresses the navigation issues that may arise when using SCORM content in different learning management systems (LMSs).

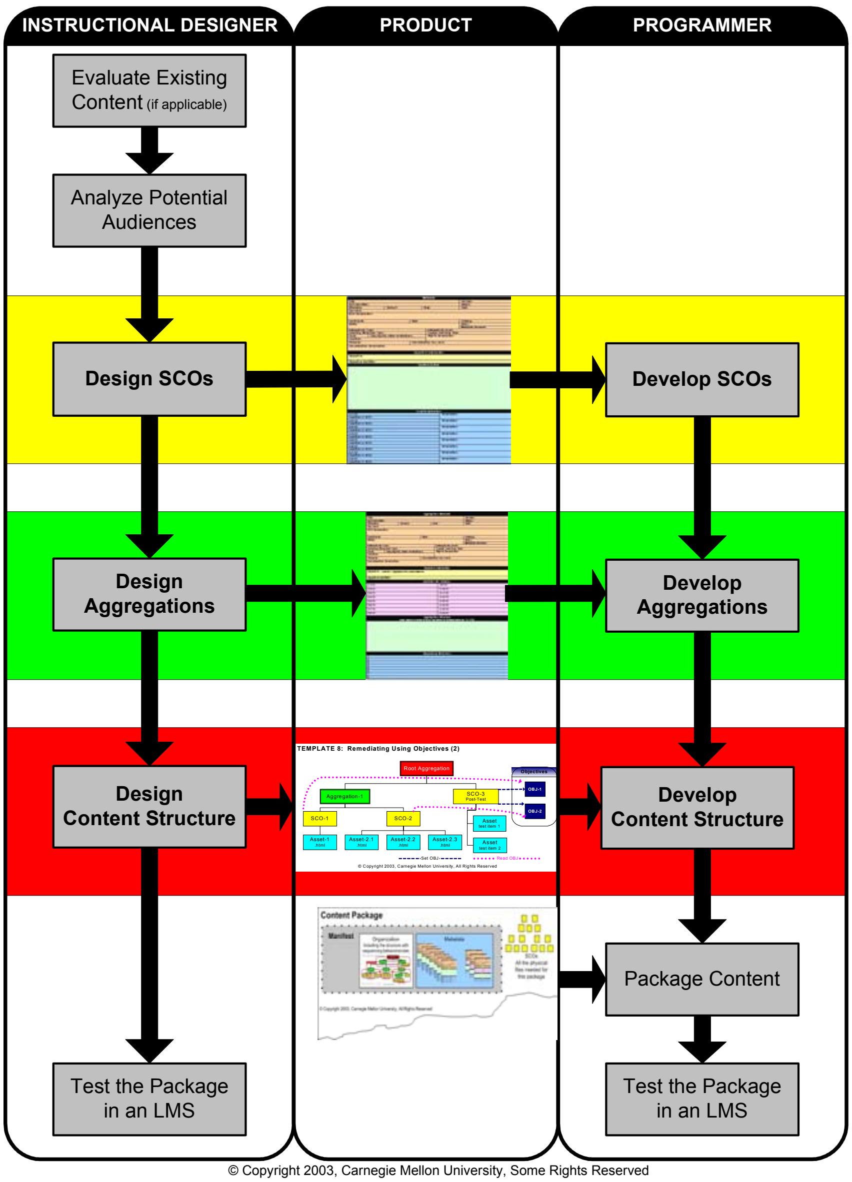
Section 7 **Sequencing Your Content**
This section explains Simple Sequencing by using templates and models that instructional designers and programmers can use to create intentional learning strategies in SCORM.

Section 8 **Packaging Your Content**
This section explains the process to create the SCORM content packages, the last step in preparing your content for use in LMSs.

Section 9 **Appendix**
The appendix includes the Learning Systems Architecture Lab (LSAL) templates of the SCO Design Specification and the Aggregation Design Specification.

The following diagram depicts the processes and tools described in this guide and how they relate to the instructional systems design process as well as to the roles of team members in the process. As you read and use the guide, you may want to return to this diagram to see how each step fits into the overall process.

This diagram also introduces the color-coding scheme used throughout this guide. In all of the diagrams in this guide, yellow boxes represent sharable content objects (SCOs). The green boxes always represent aggregations of content. The red boxes always represent a root aggregation (organization) of content.



Overview of the Design and Development Process and Products for SCORM-based instructional materials

1 Introducing SCORM

The Sharable Content Object Reference Model (SCORM) is part of a strategy called the Advanced Distributed Learning (ADL) initiative. The primary sponsors of the ADL initiative are the United States Department of Labor, Department of Defense, and the National Guard Bureau. The White House Office of Science and Technology Policy established the ADL initiative in 1997 to standardize and modernize the way in which training and education are delivered. The ADL initiative and SCORM seek to maximize technology-based learning to generate substantial costs savings. Government, academia, and private industry from around the world support ADL and SCORM initiatives. SCORM promotes efforts in four areas: *reusability*, *durability*, *accessibility*, and *interoperability*. These areas will be addressed throughout this guide.

> **Learning Management System (LMS)**
> A Learning Management System is a software package used to administer one or more courses to one or more learners. An LMS is typically a web-based system that allows learners to authenticate themselves, register for courses, complete courses and take assessments. The LMS stores the learner's performance records and can provide assessment information to instructors. A learning management system may also support the following functions: authoring, classroom management, competency management, knowledge management, certification or compliance training, personalization, mentoring, video conferencing, chat, and discussion boards.

Perhaps the easiest way to understand the need for SCORM is with an example. Many career fields require individuals to complete some type of hazardous materials training. For example, truck drivers, firefighters, and military and environmental personnel all require the training. Typically, each organization with these particular career fields would design its own hazardous materials training or pay another entity to design training for it. This practice resulted in many different organizations paying to develop redundant hazardous material training for audiences that were only slightly different. Organizations would invest thousands of dollars developing the training.

Depending upon the selected delivery format for the training, different organizations would use different delivery media (videodiscs, CD-ROMs, slide shows, etc.), different learning management systems (LMSs), different operating systems, and different authoring systems. If an organization upgraded or changed any of its information technologies, the training might not operate on the new system. Thus, the delivery format of the training would have to be revised for the new system. A change to one component in a course could impact all of the course components and require the creation and release of entirely new media.

Suppose that fifty percent of the total hazardous material training required by these organizations was generic to any audience and that this training could be used by any of the organizations. Such training could include details about each type of hazardous material, the effects of the material on humans, protective gear required to handle the material, etc. Assume that the cost to develop this portion of the training is $10,000 and that each organization spends another $10,000 to develop the portion of the training customized for its audience. If there are ten trucking

companies, ten fire departments, four branches of the military and ten environmental organizations needing the training, then these 34 organizations would spend a total of $680,000 for hazardous material training ($20,000 each).

34 organizations x ($10,000 generic + $10,000 customized) = $680,000

If there were a way that all 34 organizations could share the generic training, then together they could potentially save $330,000 by having only one organization develop the generic portion of the training (at a cost of only $10,000) and then share it with the others.

$10,000 generic + (34 organizations x $10,000 customized) = $350,000
TOTAL SAVINGS = $330,000

From this example, it is obvious that hundreds of different organizations around the United States, not to mention the world, might have redundant training requirements. In this example, the total cost savings generated by sharing hazardous materials training would be even greater than estimated.

Unfortunately, another factor limits the viability of this solution. Despite the substantial cost savings that would result from having one organization develop the training for the 34 other organizations, each organization likely uses a different learning management system, uses different hardware, and uses a different authoring system. So, what worked technologically with the hardware and software of the developing organization might be useless on another system. The training resources could not be *reused* because they were not *interoperable* with all the different systems of the other organizations. The goal of SCORM is to create flexible training options by ensuring content that is *reusable, interoperable, durable,* and *accessible,* regardless of the content delivery and management systems used.

SCORM Concept	Definition	Example
Reusable	Content is independent of learning context. It can be used in numerous training situations or for many different learners with any number of development tools or delivery platforms.	Content developed by a refinery to train its employees to respond to a petroleum spill could be reused by the fire department as part of a hazardous materials training program.
Interoperable	Content will function in multiple applications, environments, and hardware and software configurations regardless of the tools used to create it and the platform on which it is delivered.	Content developed in one authoring system where the delivery platform is a CD on a non-networked Macintosh will also operate over the Web on a PC using both Internet Explorer and Netscape equally well.
Durable	Content does not require modification to operate as software systems and platforms are changed or upgraded.	Upgrading an operating system from Windows NT to Windows 2000 has no impact on the delivery of content to the learner.
Accessible	Content can be identified and located when it is needed and as it is needed to meet training and education requirements.	A manager can conduct an online search for training on sexual harassment and identify appropriate materials for her specific organizational needs based on information provided in the content metadata.

Table 1.1: SCORM Concepts and Definitions

SCORM achieves *reusability, interoperability, durability,* and *accessibility* with the use of sharable content objects (SCOs) composed of assets that launch in a SCORM run-time environment. Metadata enables managers, learners, designers, programmers, and others interested in education and training to identify and locate instructional materials and assets using tools such as an online content repository.

Metadata
Metadata is "data about data." It is the information that describes what your content is, both the individual pieces (the assets and SCOs) and the content packages. Metadata enables instructional designers searching for content or assets to locate them with relative ease and determine whether they will be useful before downloading or requesting rights to your SCOs or assets. See Section 3.4, Locating Your Instructional Materials for more information on selecting and completing metadata fields.

This Page Intentionally Left Blank

2 Defining Sharable Content Objects (SCOs)

It is important to understand that the term sharable content object (SCO) means different things to different people. Depending on your specific training and education needs, your SCOs may consistently be large or small, or they may vary depending on the type of instructional materials you are creating. The composition of your SCOs may vary as well. They may contain single learning objectives, collections of learning objectives, tests, scenarios, simulations, etc.

The term SCO also means different things to instructional designers than it does to programmers. Instructional designers, authors, and content developers focus on the actual instructional material in the SCO, so for them, a SCO is content. For a programmer, a SCO may merely be the pointer to the actual SCO file when creating the manifest, or a SCO may be a combination of SCO files and metadata when creating the content package.

In addition, you will hear terms such as learning object (LO) and reusable learning object (RLO) used almost synonymously with SCO, but also defined differently than SCO is defined here. It is important for your organization to define and adhere to your own unique interpretation of a SCO for your SCORM implementations. This guide provides a somewhat flexible definition of a SCO. It also provides instructional design guidelines and sequencing models that will assist you in creating an effective instructional design that complies with SCORM while meeting the needs of your specific organization.

2.1 Assets

Assets are electronic representations of media, text, images, sounds, Web pages, assessment objects, and other pieces of data that can be delivered to a Web client. Assets, like the sharable content objects (SCOs) in which they appear, are highly reusable. In order to be reused, assets are described using metadata so that they are both searchable and discoverable in online repositories.

Assets may be reusable in many contexts and applications. A digital photograph of a Himalayan cat may be catalogued in an online repository by an animal shelter in New York. When creating an online course for veterinarians on different breeds of cats, a university in Iowa might search the repository, identify the photograph, and utilize it in their LO. An online animal reference guide designed for public reference could also use the asset.

2.2 Sharable Content Objects (SCOs)

A sharable content object (SCO) is a collection of assets that becomes an independent, defined piece of instructional material. SCOs are the smallest logical unit of instruction you can deliver and track via a learning management system (LMS). Additionally, under the current SCORM, SCOs cannot directly access other SCOs. So, you could not create a SCO with any links to

content in other SCOs. Put another way, this means a learner cannot access supplemental content from another SCO. It is very important to remember that each SCO should be able to stand alone.

With SCORM all inter-SCO sequencing is directed by the LMS through hard-coded sequencing behaviors. This is significantly different from the way most CBT lessons and courses function.

You can view a SCO as any "traditional" instructional design component such as a lesson, a module, a unit, a segment, or a course. As a result, you can use them in several different ways. The way you use the SCO will depend on the level at which you want to track the learner, as well as the type and structure of your specific content. Here are some of the "roles" a SCO can play in your instructional materials:

1) Learning objectives (the SCOs) in a lesson
2) Segments (the SCOs) in a lesson
3) Lessons (the SCOs) in a module
4) Modules (the SCOs) in a course
5) Lessons (the SCOs) in a course
6) Units (the SCOs) in a course

A SCO is independent of other instruction, so it cannot rely on other SCOs or a particular course structure to give it meaning or place it within a certain context. For instructional designers, this may pose a concern: How do you ensure the instructional integrity of a SCO if there is no supporting course structure and you don't know the context in which it may be used?

If you are an instructional designer, author, or content developer accustomed to thinking in traditional instructional design terms (i.e., courses, units, segments, lessons, and modules), then the easiest way to think of a sharable content object might be as a stand-alone lesson or a single learning objective. Effective completion of the SCO will impart the knowledge or skill for which it was designed. Since SCORM says SCOs should be small enough to allow reuse across multiple learning contexts, as a best practice, a SCO should represent at least a single instructional objective and all of the related materials required to support that objective. As such, a SCO should be instructionally sound. Remember, no matter what instructional strategy you employ or delivery medium you use, your content will only be as effective as the objectives you write. Ensure you spend adequate time, in advance of the technical implementation, determining the types of objectives you want to use for your content, the granularity of each objective you write, and the conditions and standards your objectives include.

The SCO should contain the complete instructional content and all assets supporting the instruction. As a best practice, a SCO should also contain any tests required to sample existing or attained knowledge or skills (pre- and post-tests or embedded knowledge "checks" or quizzes).

However, if deemed appropriate, tests can be created as individual SCOs separate from the instruction. This would enable content-only SCOs to be used both with and without tests. Section 5, Structuring Tests in SCORM, discusses additional considerations for testing in SCORM, and Section 7, Sequencing Your Content, shows examples of content structure diagrams you can use to achieve the various types of test structures and tracking you require.

A sharable content object may also contain various sub- or enabling objectives that support the primary or terminal objective. However, the LMS is only required to track at the level of the SCO, and not any smaller units inside it. If an enabling objective warrants additional tracking in the LMS, it must become a distinct SCO.

2.3 SCOs and Reusability

A well-designed sharable content object should serve numerous audiences in achieving multiple outcomes, across many contexts, making it ideal for courses and uses in addition to the ones for which it was originally designed. A sharable object may be any size, but since sharable content objects are inherently small, they are not only reusable in more contexts than a traditional course would be, but they are also easier to maintain and update as content requires changes or customization. Since the SCO is delivered via an LMS, it can also be configured in many different ways to meet many different needs.

> **Content Repository**
> A content repository is a software package designed to manage content in the form of text files, images, etc. throughout its lifecycle, including authoring, versioning control, and distribution of the content. A content repository typically includes the ability to attach metadata to its assets or content and to search for assets or content based on their metadata. A content repository also typically includes security services that allow access to assets by authorized individuals. Content repositories may be distributed to reduce the costs of acquiring large-scale storage devices and to protect data in the event of a disaster.

For example, a sharable content object called "Types of Apples" could be used, as in Diagram 2.1, for several different courses created by any number of different organizations to meet their needs. In Diagram 2.1, the "Types of Apples" SCO in Content Repository 1 is being accessed by all three of the courses shown. In the "Growing Fruit Trees" Course, the "Types of Apples" SCO is being used to show the products resulting from different types of apple trees. In the "Guide to Produce" Course, the same SCO is being used by grocers to show the complete spectrum of fruits that are available to them and to their customers. In the "Preparing Fresh Produce" Course, the "Types of Apples" SCO is used to help the cook determine the type of apple appropriate for the different dishes he prepares.

The three courses could be assembled by three different organizations around the world. Additionally, the SCOs in the repositories could have been generated by separate organizations or by the organization that owns each particular content repository. All of the SCO-generating organizations could be unrelated to the three companies creating the courses. While Course 1 is only pulling SCOs from Content Repository 1, Courses 2 and 3 are pulling SCOs from both Content Repository 1 and Content Repository 2.

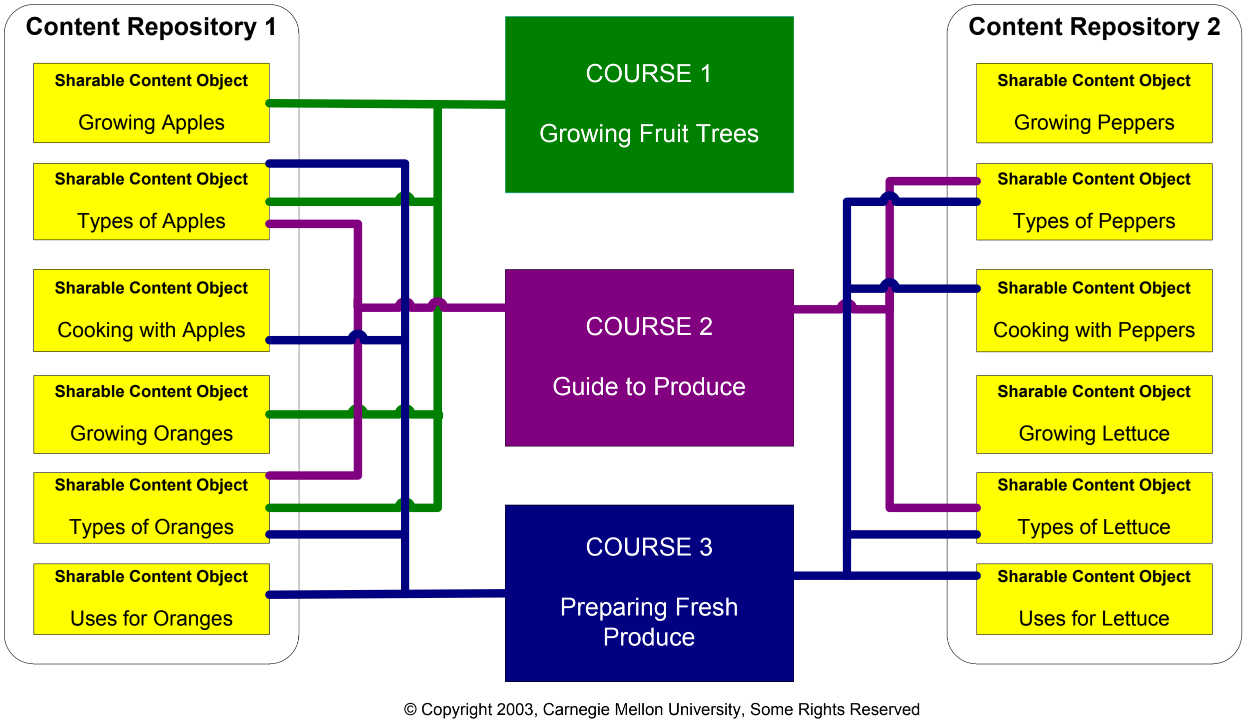


Diagram 2.1: Multiple courses created from sharable content objects (SCOs) located in different repositories.

This ability to reuse sharable content objects for many different purposes can generate significant time and cost savings. When an organization discovers an education or training need, it can search an online content repository for existing instructional materials. The organization can then retrieve content created by different organizations that is stored in different locations and configure or sequence the content to meet its own specific training needs. This "custom" course can then be delivered by a SCORM-compliant learning management system when it is needed ("just-in-time training") without waiting for weeks or months of development.

Sequencing
Sequencing is similar to the CBT term "branching" in that it describes and prescribes the manner in which the learner receives the content. In CBT, much of the branching or sequencing occurred within a lesson or course as the learner completed different tasks. However, in SCORM, the LMS sequences all activities between the sharable content objects (SCOs) and the learner, essentially performing all of the sequencing of the content based upon rules created by the designer.

3 Planning for SCORM

> **Style Guide**
> A style guide is the set of established criteria, processes, and procedures a team follows throughout the process of creating instructional materials. It should serve as the handbook or primary reference material for most questions concerning design, layout, and standardization that arise during the content development cycle. Items addressed by the Style Guide may include standardized file naming conventions, required metadata fields, a discussion of the production process, roles of team members, job aids for programs or procedures the team will follow, and the quality control procedures and checklists the team will use. The Style Guide should be minimally impacted by SCORM-specific conventions.

Since both the sharable content objects (SCOs) and assets are intended to be reused in SCORM, following standard conventions is an important factor in facilitating your design and development process, particularly as designs are passed to developers and programmers for SCORM packaging. Standardization also simplifies the creation of metadata.

3.1 Building Your Development Team

A strong and well-balanced development team with clearly defined roles and expectations will be crucial to the success of SCORM content development projects. Instructional designers, authors, and content developers should not feel as if they should understand all of the technical nuances of a SCORM implementation. This guide should provide them with the basic knowledge they need to design and author instructional materials that are SCORM-compliant. Their primary responsibility should remain developing effective instructional materials that *work within the evolving technical standards.* Programmers and system developers should focus their efforts on the technical implementation of SCORM. They should advise the instructional designers on the technical constraints that govern how SCOs are created so that the content functions optimally both technologically and instructionally. It is essential that the two groups work together from the initial planning stages of the project through project delivery. Table 3.1 outlines the responsibilities of each of the team member roles mentioned above. Your organization may refer to these roles using different names.

As design-related questions such as *"Can I do this with the content and still make it SCORM compliant?"* arise, instructional designers, authors, and content developers will rely on the technical expertise of programmers and system developers to explain the technical precepts under which they are working. Likewise, as programmers and system developers begin the technical implementation of the instructional materials, they will need to confer with designers, authors, and content developers to ensure that the materials are functioning in a SCORM-compliant LMS as the designer intended.

urn:x-lsal.cmu.edu:1sted.scormcdg b040ea31-f7c7-40b6-a43f-39db46638296

Development Team Roles and Responsibilities	
Team Member Role	**Responsibility**
Instructional Designer (ID)	Conducts the needs-and-audience analysis. Designs basic content structure. Creates design documents and design specifications and initial content outlines. Works with other team members to ensure instructional integrity of final materials.
Subject Matter Expert (SME)	Works with IDs and authors/developers to ensure the instructional materials are technically accurate and appropriate for the audience in accordance with client needs and requests. May work directly for the client.
Content Author/Developer	Takes basic instructional design and content outline from ID and researches and writes all instructional text or scripts. May request or design assets.
Programmer/Developer	Responsible for creating the content package required by the design in accordance with SCORM guidelines. May program the LMS. Ensures LMS functions properly before delivering instructional materials to students. Works closely with instructional designers to ensure content structure and sequencing behaviors meet SCORM technical implementation guidelines.
Graphic Artist/Media Producer	Responsible for creating assets (graphics, video, animation, etc.) and interfaces in accordance with requests from IDs and content authors/developers.

Table 3.1: Development Team Roles and Responsibilities

3.2 Naming and Storing Your Files

During the design and development process, designers, authors, content developers, programmers, system developers, subject matter experts, and quality control personnel handle numerous files of varying types and sizes. If you do not have a good content management system, then creating a consistent convention for all file names will make it faster and easier for all parties to identify and locate the files they need during the development process. At a minimum, a file name should include the name and number of the “course” for which it was created, the module or lesson number, a description of the item, and a version/revision number. The names should be as intuitive as possible, so that developers and programmers can quickly and easily identify files without having to view the contents of each file to ensure they are using the correct file.

Examples of Standardized File Naming Conventions	
cs101_0207_mov-angry.mpg Describes Customer Service Course 101, a movie about dealing with angry customers	**cs101_0207_txt_angry.html** Describes Customer Service Course 101, SCO number 02.07, text about dealing with angry customers
sales_coldcall_txt_leads.html Describes a Sales Course SCO for the Cold Calling content, text about how to get sales leads	**sales_coldcall-sco-spec.doc** Describes a Sales Course SCO for Cold Calling content. This is the SCO design specification.

Table 3.2: Examples of Standardized File Naming Conventions

In addition to standard file names and conventions, ensure that, during the development process, files are stored in a central location (such as on a common server), accessible to the entire production team. The folders where the files are stored should also have standard names and structures. This enables team members to quickly and easily locate specific items for any of the courses or projects on which they are working. It also eliminates emailing files back and forth and having redundant files stored on individual team member's computers. Once development is complete, the metadata defined during the design process can be applied to the asset, SCO, or spec, and the files can be archived or moved to a content repository.

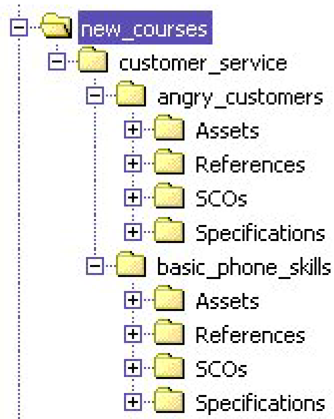


Diagram 3.1: Sample file storage hierarchy for development of multiple-course projects.

In this example, each course has a unique file directory with a standardized set of subdirectories. With standardized directory structures and file names, any team member can locate a given file for any course at any time. If anyone on the team has concerns about other individuals accessing or changing commonly stored files, consider using a content management product that enables version control, allows the "owner" of the files to set access levels and track access to the file, and requires the individual accessing the file to "check out" the file. There are a number of these products available, and they can be easily customized to suit your team's needs.

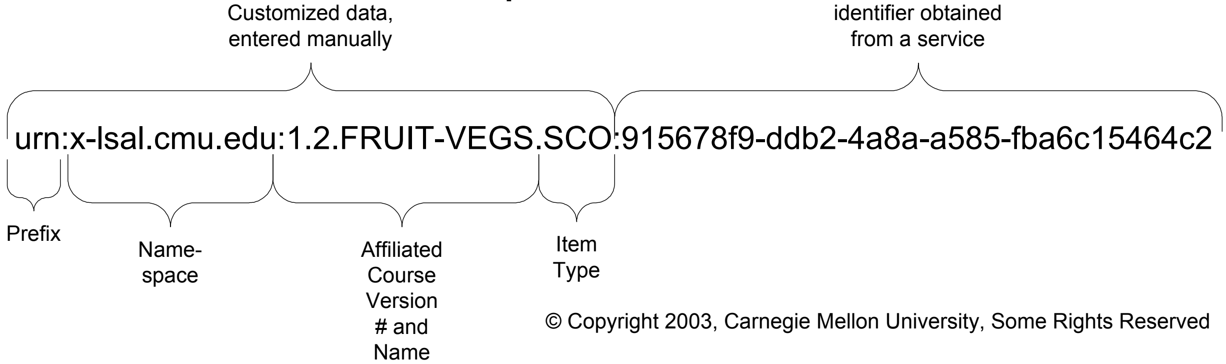
3.3 Identifying Your Materials

A globally unique identifier (GUID), like the international standard book number (ISBN) used on books, is one way of tracking objects as they are created and later identifying and locating them when they are stored in a content repository. A GUID should be included in your SCO metadata in the *catalog* and *entry* fields. The GUID on a single instance of an item should never change, so if you make modifications to your SCO (for example, version changes or updates), you will need to obtain a new GUID for each iteration of the SCO. Assets may also have GUIDs assigned to them.

GUIDs can be obtained from a number of services. However, a GUID by itself is just an alpha-numeric scheme. It will have no inherent meaning to you or anyone else. In order to give some logical meaning to your GUID, create a scheme for customizing it. Customizing your GUIDs

will ensure that you are able to locate your own materials quickly and easily. The customization of a GUID may require a manual process that you apply once you have obtained the basic GUID. The information you choose to put into the GUID structure should be standardized in accordance with your Style Guide. Below is one suggested structure for a GUID.

Sample GUID Structure



3.4 Locating Your Instructional Materials

While creating standardized file naming conventions and using GUIDs help you to track and locate your materials, SCORM has an additional requirement for the tracking and locating of materials: the use of metadata. All assets, SCOs, aggregations, and content packages require metadata in order for others to search for and locate your content.

The primary purpose of metadata is as a tool to enable those seeking assets and content relevant to their requirements to locate them more efficiently and effectively. Metadata is much like the card catalog system in a library. Using the card catalog, you could find a library resource by using a subject, author, or title card. Each card, whether subject, author or title card, contains the name of the resource, the author, the publisher, the date it was published, the Library of Congress number, and the location of the resource within the library. Metadata is the electronic card catalog for your instructional materials.

The metadata schema used in SCORM is based on the Standard for Learning Object Metadata, IEEE-SA Standard 1484.12.1-2002 also called LOMv1.0. LOM as well as other metadata schemas can also be used to help you manage your content more effectively. You can use it to assign digital rights, describe file formats used to install and store it, and track and update version and revision status. LOMv1.0 does not require the completion of many fields for compliance. However, SCORM does require the use of certain categories and fields of metadata.

SCORM uses nine categories of metadata fields called elements. There are several sub-categories in each field. This results in about 70 possible metadata fields. Since the metadata field options are numerous, your Style Guide should define which of these fields will be required for your projects and which fields will be optional. The metadata required for SCOs and aggregations is identical, however assets have different metadata requirements in SCORM.

Metadata should only reference one item, so if you make multiple versions or revisions to an item, each iteration should have its own unique metadata. You can accomplish this by using different globally unique identifiers and the version field in the Life Cycle element. Remember that your GUID will be divided into two separate metadata fields in your metadata record: *catalog* and *entry*. Specific information about metadata fields is located in Table 3.4, Recommended Metadata Fields for SCOs and Aggregations.

Relating a GUID to Metadata Fields

Catalog: x-lsal.cmu.edu
Entry: 1.2.FRUIT-VEGS.SCO:915678f9-ddb2-4a8a-a585-fba6c15464c2

urn:x-lsal.cmu.edu:1.2.FRUIT-VEGS.SCO:915678f9-ddb2-4a8a-a585-fba6c15464c2

When creating your metadata, always remember the knowledge base, search habits, and search vocabulary of those seeking your content, and develop your metadata with these things in mind. This includes using the most specific and descriptive terms possible.[1] For example, Table 3.3 shows how you could complete some metadata fields if this guide were to be used as a SCO.

Metadata			
Title: SCORM Best Practices Guide for Content Developers			**Version:** 1st Edition
Keyword: SCORM, content developers, ADL, SCORM metadata, simple sequencing			**Status:** Final
SCO Description: The SCORM Best Practices Guide for Content Developers describes processes and procedures for designing and developing SCORM-compliant content.			**Date:** 2003-02-27
SCO Catalog: x-lsal.cmu.edu			**Structure:** Linear
SCO Entry: 1sted.scormcdg.b040ea31-f7c7-40b6-a43f-39db46638296			**Format:** portable document format (.pdf)
Classification Description: SCORM Content Development			**Size:** 8400000 bytes
Purpose: Discipline	**Classification Keyword:** SCORM Content Development		
Contribute: Learning Systems Architecture Lab	**Role:** Author		**MD Scheme:** LOMv1.0
Entity: Learning Systems Architecture Lab, Carnegie Mellon University; Pittsburgh, PA, USA; www.lsal.cmu.edu		**MD Catalog:** x-lsal.cmu.edu **MD Entry:** 1sted.scormcdgmetdata.e7054c26-92df-4dbb-9acb-afd9bf7a9ca3	
Learning Resource Type: Narrative Text		**Interactivity Type:** Expositive	
Cost: No	**Copyright & other restrictions:** Yes	**Interactivity Level:** very low	
Rights Description: Redistribute or reprint is allowed. Credit must be given to Carnegie Mellon University LSAL. May not be used for commercial purposes.		**Typical Learning Time:** 4 hours **Location:** http://www.lsal.cmu.edu/lsal/expertise/projects/developersguide/index.html	

Table 3.3: SCO Metadata for SCORM Best Practices Guide for Content Developers

To be optimally accessible and locatable, your metadata should include the elements in the Recommended Metadata Fields table that follows. However, your team must decide which specific fields are most appropriate for the type and volume of content you are creating. "A few

[1] CanCore Learning Object Metadata, Metadata Guidelines, Version 1.1, p. 0-9.

well-chosen and well-implemented metadata elements will enhance resource discovery in a cost effective manner; the more elements implemented, the greater the chance for error and the higher the cost for a decreasing return on investment with respect to resource discovery."[2]

Since your developers and programmers will need to include the metadata as XML when they package your content, it is important to use a common tool that is useful for both programmers and designers. The Style Guide should provide this standard format in the form of a SCO design specification or a design form, that specifies only the metadata fields your team requires (see the SCO Design Specification Template in the Appendix for one possible format). Your Style Guide should also assign responsibility for defining the metadata fields. In most instances, the instructional designer or developer should define the metadata fields for the programmer in advance.

A SCO design specification could be created as an interactive tool that would allow you to automatically populate fields, create links to services that populate fields, or use pull-down menus to populate fields. Pull-down menus will work particularly well for the elements that use vocabularies. In addition, several commercially available software products now provide users with SCO design templates that automatically generate metadata fields when the asset or SCO is created.

[2] CanCore Learning Object Metadata, Metadata Guidelines, Version 1.1, p. 0-10.

Recommended Metadata Fields for SCOs & Aggregations			
Element Name	**Required by SCORM**	**Explanation**	**Example**
1 GENERAL	Y	"Traits that provide a reasonable first point of contact with a learning resource"[3]	This is summary-level element only. The fields you complete are below it.
1.1 Identifier	Y	A globally unique label that identifies the SCO. It should be machine generated to ensure uniqueness.	This is roll-up element only. The fields you complete are below it.
1.1.1 Catalog	Y	The name of the catalog systems being used. This is the existing library or catalog system used to classify the SCO. You may also chose to create a catalog system of your own.	Library of Congress, ISBN, DOI, Dewey Decimal, etc.
1.1.2 Entry	Y	The unique value or entry in the catalog listing used to locate and access the SCO. This entry is different than the entry used for the metadata.	1.0.pie.sco.915678f9-ddb2-4a8a-a585-fba6c15464c2
1.2 Title	Y	This should be the actual title of the content of the resource.	Growing Avocado Trees
1.4 Description	Y	Succinct textual description of the resource using primarily keywords.	Provides techniques for growing three species of avocado trees in several climates.
1.5 Keyword	Y	Textual words or phrases used to identify or define the resource.	Growing avocado trees, evergreen fruit trees, West Indian Avocados, Mexican Avocados, Guatemalan Avocados
1.7 Structure	N	The manner in which the actual resource (content) is organized, formatted, or structured.	*Uses existing IEEE LOMv1.0 vocabulary value.*
2 LIFE CYCLE	Y	The history and current state of the resource, as well as those working on the resource.	This is summary-level element only. The fields you complete are below it.
2.1 Version	Y	The current edition or iteration of the resource.	Version 2.0
2.2 Status	Y	The current state or condition of the resource.	*Uses existing IEEE LOMv1.0 vocabulary value.*
2.3 Contribute	N	Describes those working on the resource (people and organizations).	This is a summary-level element only. The fields you complete are below it.
2.3.1 Role	N	The type of contribution made.	*May use existing IEEE LOM vocabulary or best practices.*
2.3.2 Entity	N	The group or organization that contributed to the resource (most relevant first).	See the guidelines in Section 2.2, *Metadata*, of *The SCORM Content Aggregation Model*, since the entry in 2.3.1 *Roles* impacts the entry here.
2.3.3 Date	N	The date the resource was created, submitted, or versioned.	2002-07-15; YYYY-MM-DD
3 META-METADATA	Y	Describes information about the metadata itself (who created the record, when, how, etc.).	This is summary-level element only. The fields you complete are below it.
3.1 Identifier	Y	A globally unique label that identifies this set of metadata. It should be machine generated to ensure uniqueness.	This is roll-up element only. The fields you complete are below it.
3.1.1 Catalog	Y	The name of the catalog systems being used. This is the existing library or catalog system used to classify the metadata.	Library of Congress, ISBN, DOI, Dewey Decimal, etc.

[3] CanCore Learning Object Metadata, Metadata Guidelines, Version 1.1, p 1-1.

Recommended Metadata Fields for SCOs & Aggregations			
Element Name	**Required by SCORM**	**Explanation**	**Example**
3.1.2 Entry	Y	The unique value or entry in the catalog listing used to locate and access the metadata. This entry is different than the entry used for the item itself.	1.0.pie.sco.915678f9-ddb2-4a8a-a585-fba6c15464c2
3.3 Metadata Scheme	Y	SCORM references the IEEE Learning Object Metadata (LOM) v1.0 for metadata. Some fields are required by SCORM that may not be required by IEEE LOMv1.0.	IEEE LOMv1.0. NOTE: Other metadata schemes do exist. Some examples include DublinCore and CanCore.
4 TECHNICAL	Y	Technical requirements, characteristics, and structure of the resource.	This is summary-level element only. The fields you complete are below it.
4.1 Format	Y	Technical data type and the software needed to display or access the resource.	MIME, "non-digital"
4.2 Size	N	The actual (not compressed) size of the resource in BYTES (not MB or GB).	532 bytes
4.3 Location	Y	Access point or internet location of the actual resource.	http://www.crfg.org/pubs/ff/avocado.html
5 EDUCATIONAL	N	Describes the educational characteristics or philosophies of the resource.	This is summary-level element only. The fields you complete are below it.
5.1 Interactivity Type	N	Describes the type of interaction the resource provides to the learner.	*Uses existing IEEE LOMv1.0 vocabulary value.*
5.2 Learning Resource Type	N	Describes the most dominant instructional strategy of the resource.	*Uses existing IEEE LOMv1.0 vocabulary value.*
5.3 Interactivity Level	N	Describes the amount of interaction between the learner and the resource.	*Uses existing IEEE LOMv1.0 vocabulary value.*
5.4 Typical Learning Time	N	Defines the approximate time required to complete the resource.	Refer to ISO 8601 Data Elements and Interchange Format
6 RIGHTS	Y	Describes the intellectual property rights and conditions for use of and access to the resource.	This is summary-level element only. The fields you complete are below it.
6.1 Cost	Y	Identifies whether or not a fee is charged for use of or access to the resource.	*Uses existing IEEE LOMv1.0 vocabulary value.*
6.2 Copyright and other restrictions	Y	Identifies whether or not there are copyrights or restrictions on use of or access to the resource.	*Uses existing IEEE LOMv1.0 vocabulary value.*
6.3 Description	N	Textual description or comments regarding use of or access to the resource.	For permission, contact…
9 Classification	Y	Describes the placement of the SCO in a given classification system	This is summary-level element only. The fields you complete are below it.
9.1 Purpose	Y	Describes why the SCO is being classified.	*Uses existing IEEE LOMv1.0 vocabulary value.*
9.3 Description	Y	Describes the resource according to it's classification	Prerequisite: Must complete SCO called Evergreen Trees before seeing this SCO.
9.4 Keyword	Y	Textual words or phrases used to identify or define the resource according to its classification.	Evergreen Trees, Growing Avocados

Table 3.4: Recommended Metadata Fields for SCOs and Aggregations

3.5 Using Tools to Design Your SCOs

A SCO design specification (spec) is a single document or tool you can use to centralize several aspects of the design and development process. If you are creating a design specification, it should include space for a complete outline of the text or content for each SCO, a list of all assets you will use in the SCO, and the metadata you want to have assigned to the SCO. With all of this information in one place, designers will have a central place to develop the content. Having the metadata in the same place as the instructional content planned for the SCO also makes it easier for programmers to locate assets and resources quickly and easily when packaging the content. Specs should be standardized for your team with each spec having the same information in the same location. A spec should also have standard formatting, fonts, etc. This makes it easier for the programmers to locate pertinent information when creating a package, thereby making your development process more efficient.

Each SCO will require a unique SCO design specification with unique metadata. The appendix of this guide includes a template of a SCO design specification you can customize for your SCORM projects. This template follows the SCORM best practices in this guide. The metadata fields included in the form are based on those in the table of Recommended Metadata Fields above.

Depending on your particular needs, you can create your own spec in several ways. If you work in a Web-based development and storage environment, create the spec as an online form. If you prefer to work in a word processing environment, store the files on a common development server so the entire team can access them. Again, you could create your spec as a simple document or an interactive tool that includes links to services that automatically populate information such as globally unique identifiers (GUIDs). The fields could also be represented as pull-down menus, particularly for those metadata elements that use vocabularies. If you prefer to use a software application to design and develop your SCOs, choose from several commercially available products that provide users with templates that automatically generate metadata fields when the asset or SCO content is created.

This Page Intentionally Left Blank

4 Identifying and Designing Sharable Content Objects (SCOs)

4.1 Ensuring the Instructional Integrity of SCORM Content

Many instructional designers express concerns about the reusability and context-free aspects of SCORM. Instructional designers wonder how you can maintain the instructional integrity of SCORM content when you don't know who will use it, when they will use it, or with what other materials they will use it.

As a best practice, one of the easiest ways to ensure the instructional integrity of SCORM content is to make each SCO a stand-alone "lesson" or instructional unit. Since a SCO is intended to be inherently small, it should represent a single instructional objective and all of the related materials and resources required to support that objective. Structured in this manner, the effective completion of the SCO will impart the knowledge or skill for which it was designed.

This guide will not attempt to define how you should write learning objectives: whether they should be terminal or enabling objectives, whether they should be performance- or knowledge-based objectives, etc. Keep in mind, however, that well-written objectives containing a behavior, a condition, and a criterion will most effectively determine and track whether the learner is acquiring knowledge or skills and the level at which the knowledge or skills have been acquired. Good content makes good SCORM content, so the time you spend carefully analyzing desired outcomes and designing learning objectives may make the greatest difference in your ability to create "good" sharable content objects.

This section is designed to show you a process for creating sharable content objects. The tips and techniques explained in this section will facilitate your development of SCORM-compliant training, but through your own implementation, you will continue to learn more about the ways you can create SCOs and SCORM-compliant instructional materials.

Again, since SCOs are intended to be inherently small, objectives such as "understand how a car operates" will likely be too comprehensive to become good SCORM content. In this case, you might want to consider objectives like "in accordance with a user's manual, describe the process used to change a flat tire, without assistance" or "understand the nutritional value of the avocado in a healthy diet, as defined by the US Department of Agriculture." If SCOs are limited to a single, well-written objective, then it is easier to make more of them context-neutral. Where context-specific instruction is required, you can create context-specific objectives like "in accordance with the owner's manual, describe the process used to change a flat tire on a 2002 Chevrolet Malibu, without assistance."

urn:x-lsal.cmu.edu:1sted.scormcdg b040ea31-f7c7-40b6-a43f-39db46638296

4.2 Moving from a Traditional Course Structure to SCORM

Traditional course structures tend to follow a hierarchical scheme with a course being composed of various modules and each module being composed of lessons. Each lesson then has one or more objectives. The lesson may or may not have a learner assessment. The diagram below shows a "traditional" course called Types of Fruits and Vegetables. Assume this hypothetical course was designed for growers to give them detailed information about all aspects (growing, harvesting, selling, nutrition) of different types of fruits and vegetables. There are three lessons in each module, each represented by a different color scheme. In this format, a grower who wants to learn specific information about eggplants would have to complete, at a minimum, the entire lesson on Solanaceae Vegetables to see the information on eggplants. Likewise, if the grower wanted to learn about eggplants and pineapples, she would have to see, at a minimum, both the entire Solanaceae Vegetables lesson and the entire Herbaceous Fruits lesson. This limits the ability of learners to access only the content they desire or the crucial objectives and also limits the reusability of the instructional materials.

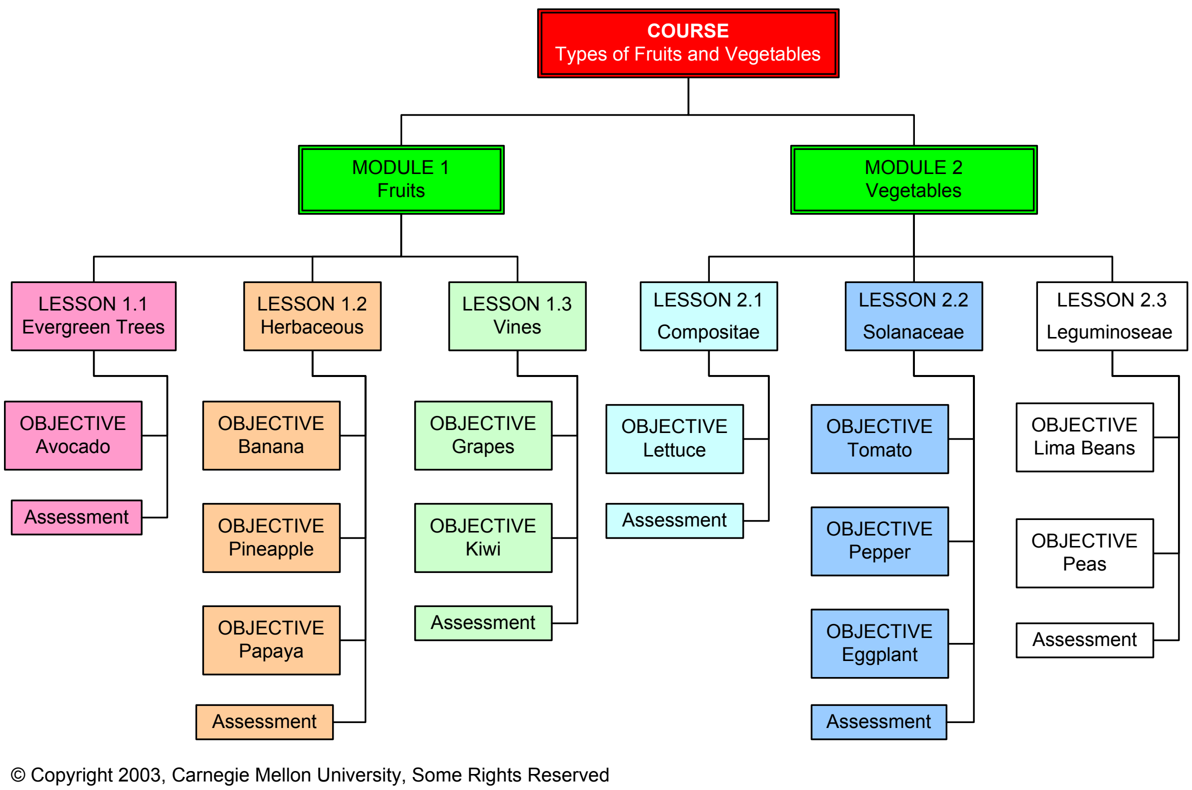


Diagram 4.1: Example of a Traditional Course Content Structure Diagram

With SCORM 1.3, you can still achieve the basic functions of a hierarchical instructional scheme like the one above by apply simple sequencing behaviors and rules to your content. Simple sequencing will be addressed in Section 7. The remainder of this section will show how to define and create SCORM content, so that both you and your learners will have far greater flexibility in defining and creating learning experiences.

4.3 Designing SCOs from Existing Instructional Materials

As SCORM becomes more prevalent, you will likely be asked to convert existing training materials to SCORM-compliant materials. The materials you are asked to convert may or may not have been developed by your organization. They also may or may not have been developed by instructional designers. Countless people have asked, "What makes good SCORM content?" The answer is good content makes good SCORM content. When tasked with converting existing materials to SCORM, it is essential to ensure the content is instructionally sound in its current form before trying to convert it to SCORM. The easiest way to do this is through a process of content "reverse engineering." Additional considerations for designing SCOs for new instructional materials are addressed in Section 4.4.

Evaluate the existing content.

Does the content teach its stated objectives? You may find, after thoughtful and unbiased evaluation, that the objectives are unrelated to the content, or the content does not teach the required objectives. If this occurs, you should determine (a) if you need to add content to teach the existing objectives, (b) if you should remove or rewrite the actual objectives, (c) if you should remove irrelevant content, or (d) if you should re-design the organization of the content. For any of these options, there may be impacts to your schedule, budget, and resources. If you are in the proposal process (proposing to complete the conversion of existing content to SCORM), then consider the impacts of these issues in your costing model since they may be significant.

Analyze the potential audiences.

One of the goals of SCORM is to create reusable content. The same content can frequently be reused in different industries by different learners to achieve different outcomes. For example, if you are designing the fruits and vegetables subject matter for growers, and one of your objectives is "identify the nutritional value of avocados," determine if nutritional information about avocados is specific to growers or if there is some nutritional information about avocados that other groups of individuals may be interested in, regardless of the context. Many times, you will find that the content can be used by more than the individuals for whom it was originally designed.

Conduct a short brainstorming session with your team with a goal of identifying three to five types of individuals who might benefit from your training. Who are all the possible audiences that might be interested in the subject matter you are converting to SCORM?

Nutritionists, chefs, and restaurateurs might all benefit from learning about the nutritional value of avocados. Save the list of audiences you identify; you will need them later in the process when you create the metadata for your SCOs.

Identify the SCOs.

Since you are designing SCOs from existing content, assume that you are planning to maintain the existing structure of your content. If your content needs to be restructured, either for instructional reasons or to adhere to SCORM, refer to Section 4.4, Designing SCOs for New Instructional Materials and Section 7, Sequencing Your Content before attempting to identify your SCOs. The content structure diagram you create may require modifications or unique SCO structures to achieve the instructional outcomes you desire.

Once you've determined who the potential audiences are, you can begin to decide how the content should be "divided" into individual SCOs to make it optimally reusable while still meeting the needs of the audience for whom is was originally intended. Assume you are working with the *Types of Fruits and Vegetables course depicted in* Diagram 4.1. SCORM says a SCO should be context neutral and should stand alone. In order to accomplish this with the *Types of Fruits and Vegetables* course, we could structure the content outside of the context of a grower.

There are certain aspects of fruits and vegetables that are of interest to numerous individuals and organizations, regardless of the context. Try to create your SCO without specific references to growers. If you'd like to create a general introduction or overview specifically for growers, then create an individual SCO specific to them, knowing it may not be as reusable as the other SCOs you created. You can also create additional SCOs to augment other audiences as needed, or assume others who wish to use your content will develop their own SCOs that will work with your SCOs.

One way to approach the creation of SCOs in a context-neutral way is to make each of the existing objectives into individual SCOs. This will not only help you ensure that as a stand-alone item your SCO is instructionally sound, it is also likely to be the solution that gives you the most flexibility when you are ready to sequence your content.

Diagram 4.2 (following) shows the individual objectives from the *Types of Fruits and Vegetables Course* (from Diagram 4.1) divided into individual SCOs, rather than created as comprehensive lessons. This diagram is not intended to show the structure of the content, but rather how you can take existing courses and lessons and create SCOs from them. Each SCO in the diagram represents one objective from the existing course structure. The SCOs can now be sequenced in any manner desired by the instructional designer. Section 7, Sequencing Your Content shows numerous ways you can structure the content from this example.

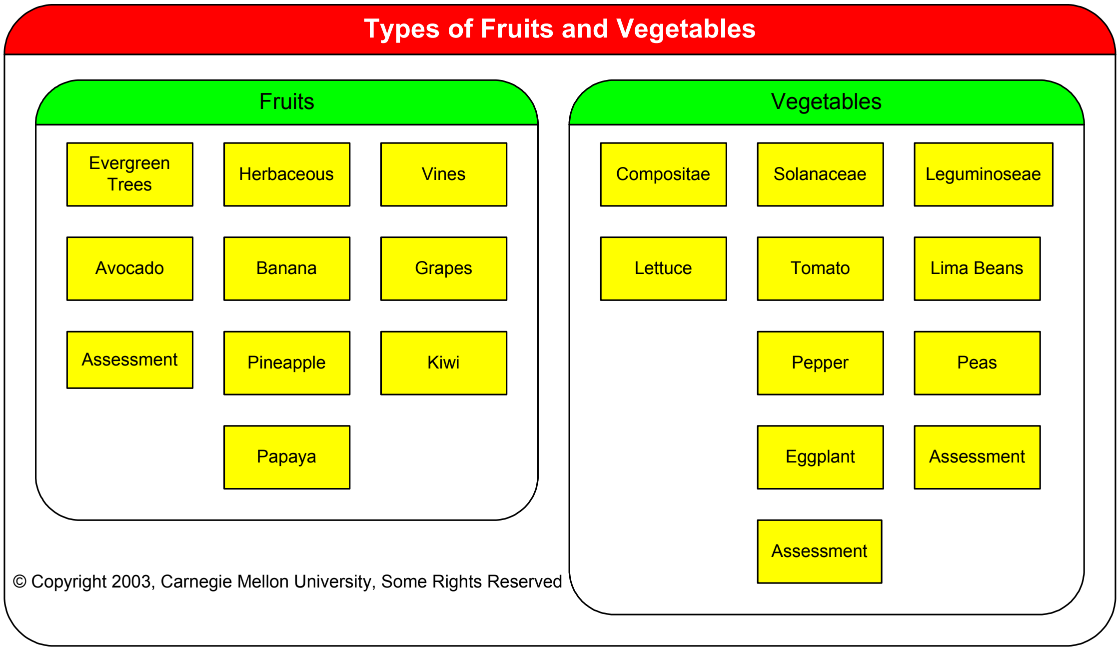


Diagram 4.2: SCOs created from the existing course depicted in Diagram 4.1

In this format, however, some of your SCOs may be too large and the content too comprehensive to meet the needs of an audience outside of growers. For example the SCO called *Avocado* might include instruction on growing avocado trees, pests that infest avocado trees, harvesting avocados, selling avocados, nutritional information on avocados, etc. These topics could possibly reach a wider and different audience. Review the content very carefully. Often the topics covered in a SCO such as Avocado can become enabling objectives that you could design as smaller SCOs, thereby making them more reusable.

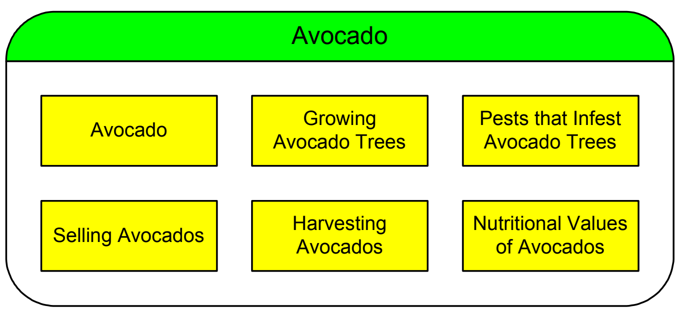


Diagram 4.3: Additional SCOs created from the existing Avocado SCO shown in Diagram 4.2

Assume the content on Avocados in Diagram 4.2 does have numerous enabling objectives. Diagram 4.3 (above) shows how you could further divide that content into SCOs that correspond to the enabling objectives. As you will learn in Section 7, this content can be

sequenced in a number of ways to meet the needs of growers as well as others interested in any of the topics covered by each individual SCO.

Develop metadata.

Once you have identified your SCOs and conducted your audience analysis, you can begin developing the metadata for your SCOs. If you are using a SCO design spec, such as the one in Appendix 9.1, you already have a tool with the defined metadata fields for your organization. For the SCO in Diagram 4.3 called *Growing Avocado Trees*, you might include the *keywords "*growing avocados" and "avocado trees." You might also include a *description* that says, "This SCO provides information about varieties of avocado trees as well as techniques for raising avocados in different climates. The content is ideal for farmers and gardening enthusiasts." The description and keywords will be useful for other individuals searching online content repositories for relevant content to use in their own courses.

<table>
<tr><th colspan="3">Metadata</th></tr>
<tr><td colspan="2">Title: Growing Avocado Trees</td><td>Version: 1.0</td></tr>
<tr><td colspan="2">Keyword: growing avocados, avocado trees</td><td>Status: Draft</td></tr>
<tr><td colspan="2">SCO Description: This SCO provides information about varieties of avocado trees as well as techniques for raising avocados in different climates. The content is ideal for farmers and gardening enthusiasts.</td><td>Date: 2003-02-07</td></tr>
<tr><td colspan="2">SCO Catalog: x-lsal.cmu.edu</td><td>Structure: Linear</td></tr>
<tr><td colspan="2">SCO Entry: 1.0.avocado.a284b779-eb39-4fd1-931a-d7e0c9af3e7e</td><td>Format: portable document format (.pdf)</td></tr>
<tr><td colspan="2">Classification Description: Avocado Trees</td><td>Size: 120000 bytes</td></tr>
<tr><td>Purpose: Discipline</td><td colspan="2">Classification Keyword: growing avocados, avocado trees</td></tr>
<tr><td>Contribute: Learning Systems Architecture Lab</td><td>Role: Author</td><td>MD Scheme: LOMv1.0</td></tr>
<tr><td colspan="2" rowspan="2">Entity: Learning Systems Architecture Lab, Carnegie Mellon University; Pittsburgh, PA, USA www.lsal.cmu.edu</td><td>MD Catalog: x-xyzorg.edu</td></tr>
<tr><td>MD Entry: 1.0.avocado.s2a0f98jqw3r09efjdsfj0iu0a98s</td></tr>
<tr><td colspan="2">Learning Resource Type: Narrative Text</td><td>Interactivity Type: Expositive</td></tr>
<tr><td>Cost: No</td><td>Copyright & other restrictions: Yes</td><td>Interactivity Level: very low</td></tr>
<tr><td colspan="2" rowspan="2">Rights Description: Redistribute or reprint in full only. Credit must be given to Carnegie Mellon University LSAL. May not be used for commercial purposes.</td><td>Typical Learning Time: 0.5 hours</td></tr>
<tr><td>Location: http://www.lsal.cmu.edu/avocado.pdf</td></tr>
</table>

Diagram 4.4: Metadata for the *Growing Avocado Trees* SCO shown in Diagram 4.3

4.4 Designing SCOs for New Instructional Materials

While it may appear easier to design new instructional materials in accordance with SCORM rather than repurpose existing materials for SCORM, the repurposing process has one advantage: you know the scope of the task since you already know what the content is, how deep the content delves into the subject matter, and how the content was intended to be structured. When designing new content to be SCORM-compliant, it will be very important to set some parameters for your design or development team.

As discussed in Section 4.3, Designing SCOs from Existing Materials, one area of analysis you'll need to consider with SCORM is the audience for whom you target your content. When considering all the possible audiences for your content, limit your brainstorming to all of the potential audiences you can identify in five to ten minutes rather than endlessly discussing the options. The process outlined here is one way in which you can work through the instructional design process to create SCORM-compliant instructional materials. You can customize this process to best fit your own team's requirements.

Determine the Instructional Strategy.

When designing SCOs for new content, one of your first tasks will be to determine what type of instructional strategy you plan to implement before you identify your individual SCOs. Because it is essential that you have a good understanding of all aspects of SCORM before you attempt to design SCOs for new content, you should read this guide in its entirety prior to beginning your design and development process.

Identify the SCOs.

Once you've determined the instructional strategy you think is most relevant to your learners, you can decide how many SCOs you will need, what content the SCOs will address, etc. You can do this in a way that will make the individual SCOs optimally reusable while still meeting the needs of the audience for whom you are designing the material. Review the guidelines in the *Identify the SCOs* portion of Section 4.3 for more considerations about identifying your SCOs, and then follow the remainder of the development process outlined in that section.

This Page Intentionally Left Blank

5 Structuring Tests in SCORM

The current version of SCORM does not address how to build, score, or structure tests, nor does it address how other specification such as the IMS Question and Test Interoperability (QTI) spec could be used to represent test items and results in SCORM. However, it does allow you to store, on the LMS, any combination of the following: score, passed/failed state, mastery status for individual objectives, and time spent on each task for each SCO. It does not address how you determine the score or the meaning of the score or mastery status. The designer and programmer must determine which variables they want the SCO to report to the LMS and how the SCO will compute the values of those variables.

Like the SCOs you create, your tests can be structured in a variety of ways to achieve a number of different outcomes. Sometimes, tests can be created as single SCOs, essentially as one object with multiple test items. At other times, to support the instructional strategy or the learner tracking you desire, you may need to create test items as individual SCOs in one or more aggregations. Remember that with the current version of SCORM, the LMS will only track at the level of the SCO. So, if you create a test with ten items and want, or are required, to know the learner's responses to specific test questions (i.e., the individual distractors selected or the pass/fail status of each distractor), then you will have to create each test item as a separate SCO within an aggregation. Creating each test item as a single SCO means that the learner will be transferred back to the LMS between each test item, creating a minor disruption in the learner's testing experience. Future versions of SCORM may address testing by providing more capabilities for capturing learner inputs and outcomes.

Below are some potential testing structures you may want to use and the instructions on how you will have to create them for your assessments to work in a SCORM-compliant manner. Section 7, Sequencing Your Content will give specific examples in the form of templates and models for ways you can structure your assessments to complement different instructional design strategies for your learners.

Creating a test as a single SCO is perhaps the easiest way to test in a SCORM-compliant system. If you choose to build a test within a single SCO, then you have the flexibility to use any commercially available authoring or testing software to design and develop the test, structure the test in any manner you desire, and determine if and how the test was completed, all internal to the SCO. The SCO containing the test will be delivered directly to the learner via the LMS. The SCO will then report the final score for the test to the LMS. If you want to link individual test items to learning objectives for remediation purposes, you can use variables called objectives in sequencing. Objectives are stored in the LMS and enable you to make sequencing decisions as shown in Section 7, Sequencing Your Content.

However, when testing within a SCO, SCORM limits the set of information that can be sent back to the LMS. As a result, your ability to collect information about how the learner performed on the test is limited. The SCO will simply report a passed/failed status or single score for the SCO to the LMS. If you are required to perform formative, summative, or operational evaluation on

your instructional materials and tests, or if you want to ensure the validity and reliability of the tests, this will not be an effective testing structure. If you simply want to "quiz" the learner to ensure they've grasped the material but have no further test tracking requirements, this may satisfy your testing requirements.

6 Navigating in SCORM

6.1 The User Interface

In SCORM, the learning management system (LMS) provides the user interface for your SCOs. This means that your SCOs will look slightly different when displayed on the screen, depending on the LMS in which they are running. Typically, the LMS interface will include forward and back buttons and a selectable table of contents. There are numerous other factors to consider when selecting an LMS. If you have access to multiple LMSs, test how your SCOs both operate and appear when running in each LMS. Do this before you publish your work for access by learners, and if possible, before you purchase an LMS.

If you've followed SCORM principles, the SCOs should operate effectively. However, depending on the color and style you selected for your SCOs, they may not appear as well as you would like them to appear. There may be contrasting color schemes, unusual fonts, etc. Some LMSs allow you to customize the appearance of the user interface to show your own logo information and color schemes. Others preclude any customization. If you plan to purchase learning management system software, consider the customization options that will be available to you before committing to a particular system.

6.2 SCORM Navigation

SCORM does not permit direct SCO to SCO navigation. This means one SCO cannot "call" or access another SCO without going through the LMS. The LMS controls all activities between the SCOs and the learner with inter-SCO sequencing, essentially performing all of the sequencing of the content based upon rules created by the designer. Intra-SCO branching (the navigation occurring inside the SCO itself) is not tied to the LMS or to the content package, so it does not constitute SCORM sequencing nor is it required to adhere to SCORM sequencing guidelines. This means that a single SCO could consist of several html pages with navigation between each page. A SCO could also be developed in an authoring system and have numerous interactions, simulations, remediations or tests within it, with all of the navigation occurring as intra-SCO branching.

Intra-SCO branching, like testing, is not addressed by current SCORM documentation. However, the numerous organizations have created guidelines and recommendations for navigational controls, audio-visual controls, and student support functions. Whichever type of navigation you choose within your SCOs, be sure to apply it across all of your SCOs for the most effective learner experience.

This Page Intentionally Left Blank

UPDATE-1 SUMMARY

Use this update to the *SCORM Best Practices Guide for Content Developers, 1st Edition* dated 2003-02-28 to replace Section 7 in its entirety.

UPDATE-1, dated 2004-04-05, supersedes Section 7 of the *SCORM Best Practices Guide for Content Developers, 1st Edition*. It replaces pages 35 - 66 of the *SCORM Best Practices Guide for Content Developers, 1st Edition* with pages numbered "UPDATE-1 Page 1" through "UPDATE-1 Page 36".

The updated Section 7: Sequencing Your Content adds a new sequencing model, Model 5, as Section 7.5.5 and corrects sequencing rules in the templates and models that changed with the release of SCORM 2004 (formerly SCORM 1.3).

For questions or concerns about changes to this document, contact the

Learning Systems Architecture Lab
Carnegie Mellon University
700 Technology Drive, Suite 3224
Pittsburgh, Pennsylvania USA 15219
Phone: +1.412.268.5322
Fax: +1.412.268.4772
Email: lsal@cmu.edu
Web: www.lsal.cmu.edu

This Page Intentionally Left Blank

7 Sequencing Your Content

In traditional multimedia and CBT, branching enabled (or sometimes forced) learners to move from one piece of content to another relatively seamlessly. Learners may or may not have known they were moving from one lesson to another or from one module to another. This was possible because robust authoring systems gave designers nearly limitless programming options for structuring and branching their content. The sequencing functionality within a lesson or between lessons, shown within the yellow box in Diagram 7.1, was hard-coded, whether based on a linear or an adaptive model.

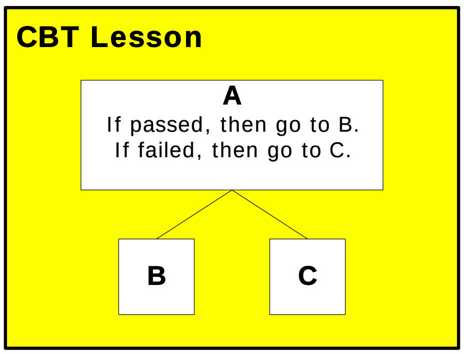


Diagram 7.1: Sequencing in CBT lessons

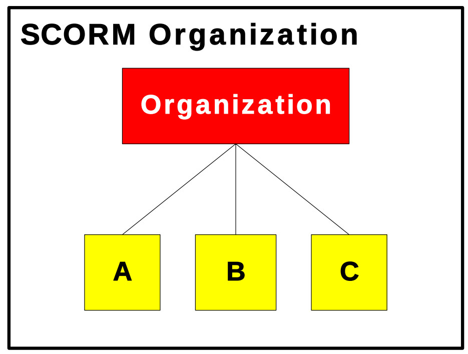


Diagram 7.2: Sequencing in SCORM 1.2

Organization
The organization is the part of a content package where SCOs are ordered into a tree structure and sequencing behaviors are assigned to them. The organization outlines the entire structure you have created for your content. The organization provides order to the otherwise unordered collection of SCOs and their metadata. Each organization has one top-level aggregation, which we refer to as the root aggregation in this document.

In the early versions of SCORM, it was not possible to sequence content in an interoperable manner. SCORM-compliant content was presented to the learner, typically as a table of contents, and learners could select the content they wanted to see. Diagram 7.2 shows a SCORM organization and several SCOs. Learners could select any SCO (represented by the yellow boxes). Instructional designers found this aspect of SCORM frustrating, since in many instances they wanted to ensure that learners would receive certain content in the order they prescribed.
To overcome this limitation, instructional designers created very large SCOs (by making several CBT lessons like the yellow box in Diagram 7.1 into SCOs like the ones shown as yellow boxes in Diagram 7.2.) Alternatively, instructional designers used the sequencing functionality provided by their in-house LMS. Neither solution worked well.

Since the goals of SCORM include interoperability and content reusability, hard-coding functionality within or between lessons made complying with the SCORM guidelines impossible:

1) Content was not interoperable when hard-coded sequencing rules were present or when sequencing rules were defined using one LMS's proprietary functionality because the sequencing functionality of one LMS could not be read by another LMS.

2) Content could not be reused when individual SCOs relied directly on other SCOs for their sequencing. Hard-coding SCOs results in one SCO "looking for" another SCO that may or may not be present. Hard-coding also limits the ability to create new or custom content structures from the same instructional materials, since each time a new structure is desired, the code attached to each individual SCO has to be updated.

7.1 Sequencing Functionality in SCORM

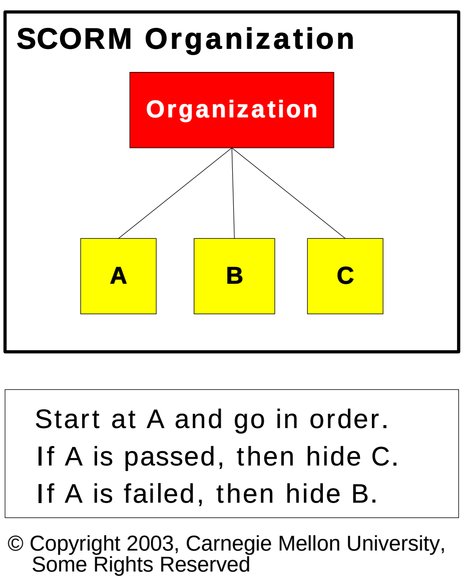


Diagram 7.3: Sequencing in SCORM 2004

SCORM prescribes nearly all functionality that occurs *outside* of the SCO itself. With the inclusion of the sequencing functionality in SCORM 2004, instructional designers have the capability to describe and prescribe the manner in which learners receive individual pieces of content from the learning management system (LMS).

The individual pieces of tracked content the learner receives are sharable content objects (SCOs). SCORM does not permit one SCO to "call" or access another SCO directly. The LMS controls the movement of the learner from SCO to SCO with *inter*-SCO sequencing. The LMS performs all of the "branching" of the content based upon behaviors defined by the designer and input by a programmer. The resulting sequencing rules get stored in the LMS as part of the manifest. This allows the same set of SCOs to be sequenced in many different ways, depending upon the designer who structures the content and the learner to whom the content will be delivered.

It is the *inter*-SCO sequencing that allows the designer to specify what is presented to the learner, when it is presented, and the attributes or functions the SCOs entail. Inter-SCO sequencing is also how SCORM allows designers to monitor and record the learner's choices and performance. All of this functionality occurs outside of the SCO itself, as shown in

Diagram 7.3, so that content can be sequenced in an interoperable manner, unconstrained by coding within the SCOs.

Intra-SCO branching (the hard-coded navigation occurring inside an individual SCO) is not tied to the LMS or to the content package, so it does not constitute SCORM sequencing nor is it required to adhere to SCORM sequencing guidelines. As a result, intra-SCO branching is not tracked by the LMS, so there is no way to report the learner's progress on individual aspects of the SCO via the LMS. However, a comprehensive score for the learner's performance on the SCO as a whole may be reported to and stored in the LMS. The scores reported to the LMS include passed/failed or a normative score between −1 and +1. Note that designers can combine intra-SCO branching and inter-SCO sequencing to create the most effective learning experiences for learners.

7.2 Preparing to Sequencing Your Content

The instructional techniques you traditionally employ may have to change slightly as you create SCORM-compliant instruction. Since the sequencing of the content is now being controlled by the LMS (which will generally be programmed by someone other than the content designer), you must carefully specify the actions and behaviors you desire for each SCO and each aggregation, all the way back to the root aggregation. If you fail to do this, the actions and behaviors of your content will be the default values defined by SCORM, which may not result in the type of learning experience you had planned or desired.

Since the sequencing in SCORM is based on a tree structure, specifying the actions and behaviors you want for your learner requires the creation of a content structure diagram. To do this, return to our example from Section 4.3, where existing content was divided into SCOs and resulted in Diagram 4.2. The SCOs that were identified in Diagram 4.2 have now been partially organized with labels as Diagram 7.4 (following).

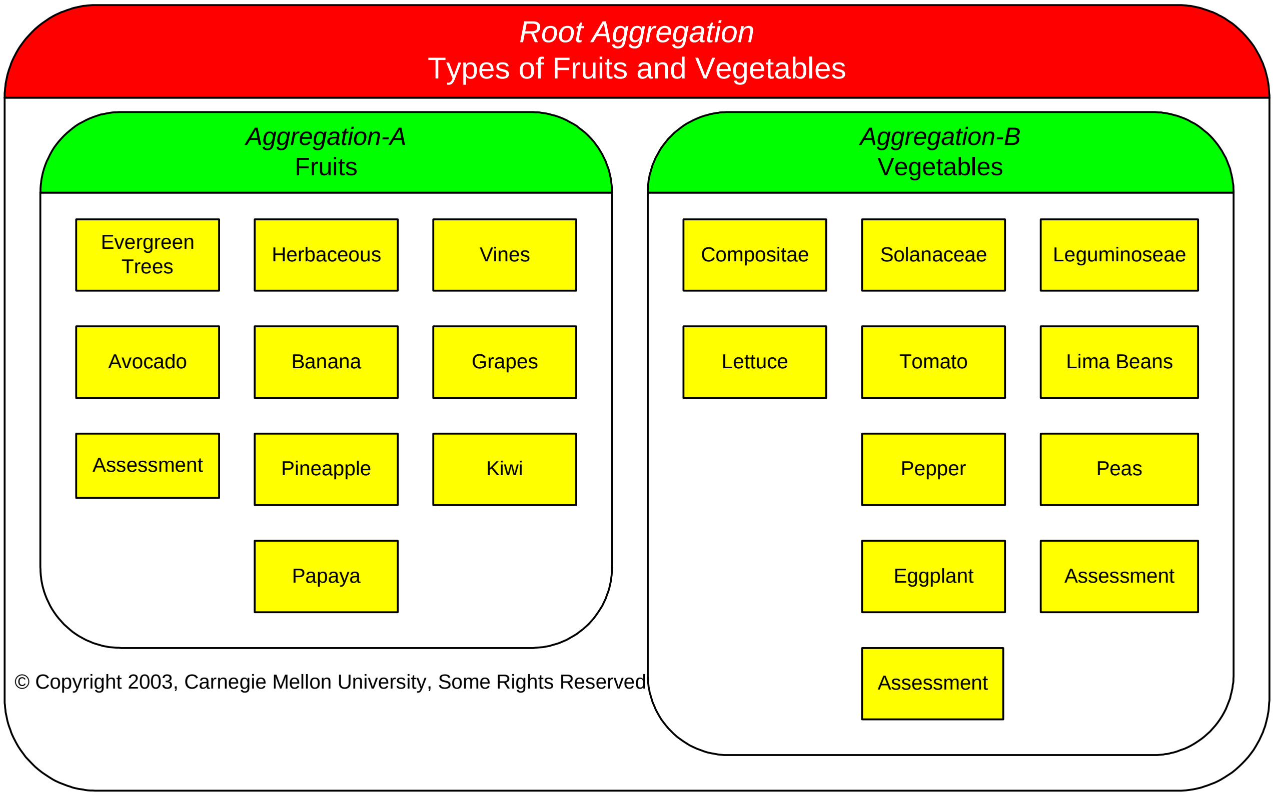


Diagram 7.4: SCOs organized for sequencing based on SCOs identified in Diagram 4.2.

So that you can quickly identify the difference between SCORM components, all of the diagrams in this guide have been created in corresponding colors. Yellow boxes represent SCOs. The green boxes above now represent what had been modules in Diagram 4.2; these boxes are now aggregations of content. The red box, previously representing the course, now represents a root aggregation.

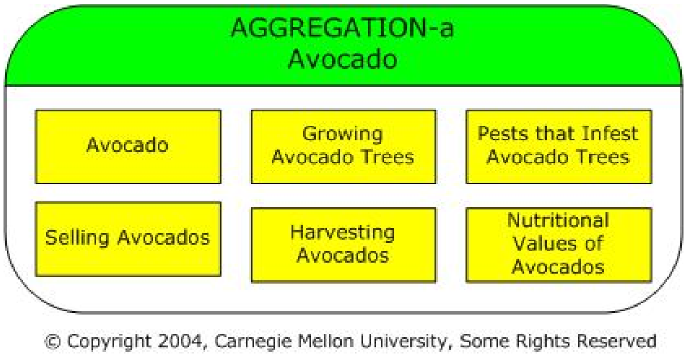


Diagram 7.5: An additional aggregation created from the *Avocado* SCO show in Diagram 7.4

Remember that in Section 4 you carefully scrutinized the Avocado SCOs you had identified and decided that some of them should be divided even further (refer to Diagram 4.3). Those SCOs will also have to be grouped before you can sequence them.

Once you have defined your SCOs, and considered some high-level groupings for them (aggregations or a root aggregation), you can begin the process of determining the content structure diagram onto which you will apply content sequencing rules. The sequencing rules (generated by your programmer) will apply the behaviors you describe for your instructional materials to ensure the instructional integrity of your content.

7.3 Understanding Sequencing Terminology

Some terms you may have used to signify a specific function of instruction may have different meanings in SCORM when you sequence your content. This requires careful use of these words, keeping in mind their definitions within the context of SCORM sequencing. One example is the word "objective" (OBJ). In traditional instructional design, an objective is used to measure the attainment of a knowledge, skill, or ability in accordance with a predefined behavior, a prescribed condition, and an achievement standard.

Objective (OBJ)
For SCORM sequencing purposes, an objective is a global variable that allows the LMS to share status values between SCOs. This gives designers greater flexibility in structuring the content under SCORM guidelines. Depending on the designer's requirements for the instruction, the objective may or may not track actual learner objectives, skills, or abilities.

In SCORM, the objective (OBJ) refers to a convenient way that a SCO can pass *MasteryStatus* parameters to the LMS. There are two types of *MasteryStatus* parameters: *PassFail* and *NormalizedScore*. You determine the criteria the SCO will use to report all the objectives' *PassFail* or *NormalizedScore* values, which will be passed to the LMS. *PassFail* simply represents whether the SCO was passed or failed. *NormalizedScore* reports a value for an OBJ to any decimal value between −1 and +1. With either of these parameters, you can choose to set their values based on a response to a single question, a complete assessment, or simply whether the SCO has actually been viewed. Each SCO can set or read multiple objectives, and a single objective can be set by or read by multiple SCOs.

Other terms with different meanings in SCORM include *complete* and *satisfied.* Traditional uses of these words would mean the learner had seen all of the content related to a given topic. For a SCO that uses the Application Programming Interface (API), you can decide the criteria that must be met for a SCO to be considered either *complete* and /or *satisfied*. For a SCO that does not use the API (a "non-communicative SCO"), the LMS will automatically set the SCO to *complete* as soon as the learner starts the SCO. As a consequence, *complete* for a non-communicative SCO does not necessarily mean that the learner saw any or all of the instructional material in the SCO. For example, the learner may have only seen the first page and then closed the SCO, thus marking the SCO complete. If you want to, or are required to, ensure the learner actually sees all of the content, then create SCOs that are single pages or do not have multiple assets.

Application Programming Interface (API)
The SCORM API is a standardized method for a SCO to communicate with the LMS when a learner is interacting with a SCO. There is a specific set of information the SCO can set or retrieve. For example, it can retrieve information, such as a student's name, or set values, such as scores.

7.4 Simplifying Content Sequencing

So that you do not have to devise a sequencing strategy from scratch for each lesson and learning experience you develop, this section of the guide provides several sequencing templates that describe potential behaviors of SCOs according to various instructional design strategies. The templates are designed to assist you in structuring your content to comply with SCORM sequencing guidelines.

Any template or combination of templates can be "overlaid" on or combined with another template, creating a more complex instructional strategy for a course or a lesson. Combining the templates provided here will give you viable sequencing models that you can adapt to meet your particular training and educational requirements. Section 7.5 shows several models for more complex instructional strategies. Depending upon how you apply behaviors to the structures, you can achieve a variety of outcomes.

These templates are not intended to be exhaustive, but they should help you begin to identify new ways in which you can construct SCORM content while adhering to sequencing guidelines, and the true intent of SCORM: creating *reusable, interoperable, durable,* and *accessible* instructional materials.

Each template section includes an introduction of the template, a content structure diagram representing the template, and the instructional strategy and sequencing rules for the template. The rules are presented in both non-technical language (called Behavior to describe what you want the learner to experience) and technical language (called SCORM Function to describe what will be coded to enable the behavior).

Designers can follow the Behaviors in the templates provided, and developers and programmers can follow the SCORM Functions to program the sequencing commands specified by the designer. In some instances, the SCORM Function says "No Unique SCORM Function" for the programmers. This occurs because the designer specifies a behavior that is either internal to the SCO or is not impacted by SCORM. Several templates include multiple applications of the rules so you will understand that identical content structure diagrams (or courses, lessons, etc.) can be sequenced in numerous ways.

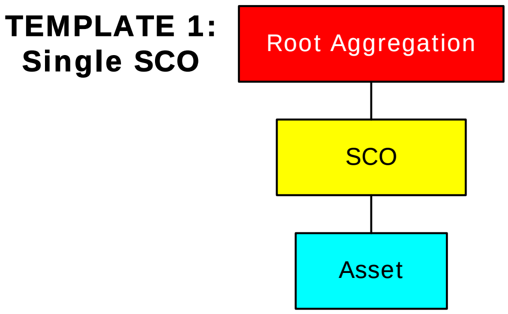
Template or Model	Description	Rule Applications
Template 1	Single SCO with a Single Asset	1
Template 2	Single SCO with Multiple Assets	1
Template 3	The Black Box; single SCO with multiple assets and complex internal structure	1
Template 4	Multiple SCOs with Assets	2
Template 5	Remediating Using Objectives	2
Template 6	Pre- and Post-Test Sequencing	1
Template 7	Pre- and Post-Test Sequencing (2)	1
Template 8	Remediating Using Objectives (2)	1
Template 9	Basic Three-way Branching	2
Template 10	Pre- and Post-Test Sequencing with New Content for Remediation	1
Model 1	Remediating Multiple Aggregations	2
Model 2	Mastery Testing Multiple Aggregations	1
Model 3	Pre- and Post-Test Sequencing with Aggregations	1
Model 4	Traditional CBT Branching with Multiple Decisions	1
Model 5	Customized Learning Using Three-Way Branching	1

Table 7.1: Summary of Sequencing Templates and Models

In this document and in the sequencing rules, we refer to halting the learning in training and requiring manual intervention by the instructor. You might want to use this type of an instructional strategy if you need to prevent the learner from seeing additional content because (1) they require face-to-face interaction with an instructor to ensure they have grasped the material, (2) they need assistance beyond that which is available in the remaining content, or (3) they will be unable to understand the remaining content without a strong understanding of the content they have completed. You can accomplish this by creating rules that result in the learner being prevented from seeing any SCO. The way in which manual intervention is implemented will vary by LMS; it is not specified by SCORM, so ensure that you carefully test this functionality before using it.

7.4.1 Template 1: Single SCO

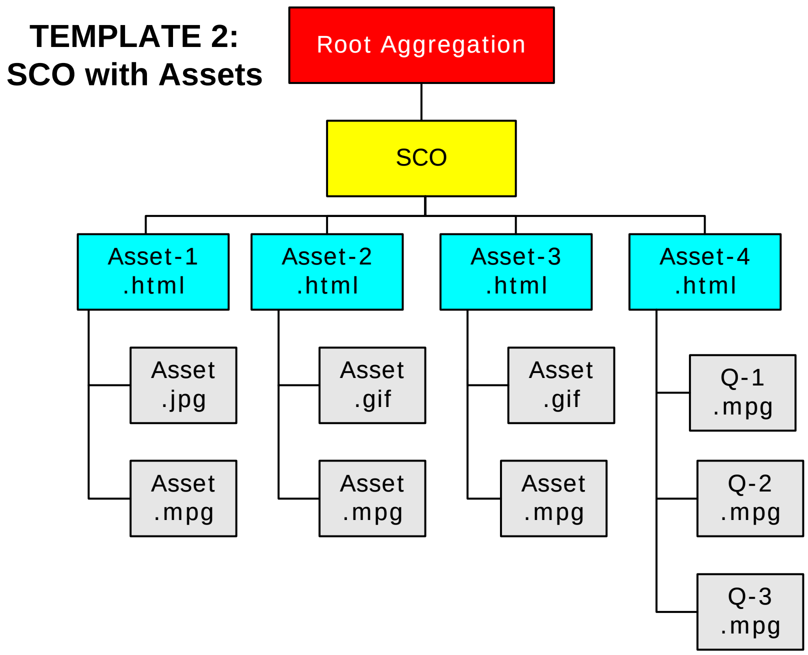
This is the most basic SCO structure. A root aggregation contains a single SCO. The SCO may be any size and have any amount of intra-SCO branching or an assessment. This SCO contains one asset.



Template 1 Rules:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must complete the SCO.	Root Aggregation Rollup: If All Satisfied, satisfied.

7.4.2 Template 2: SCO with Assets

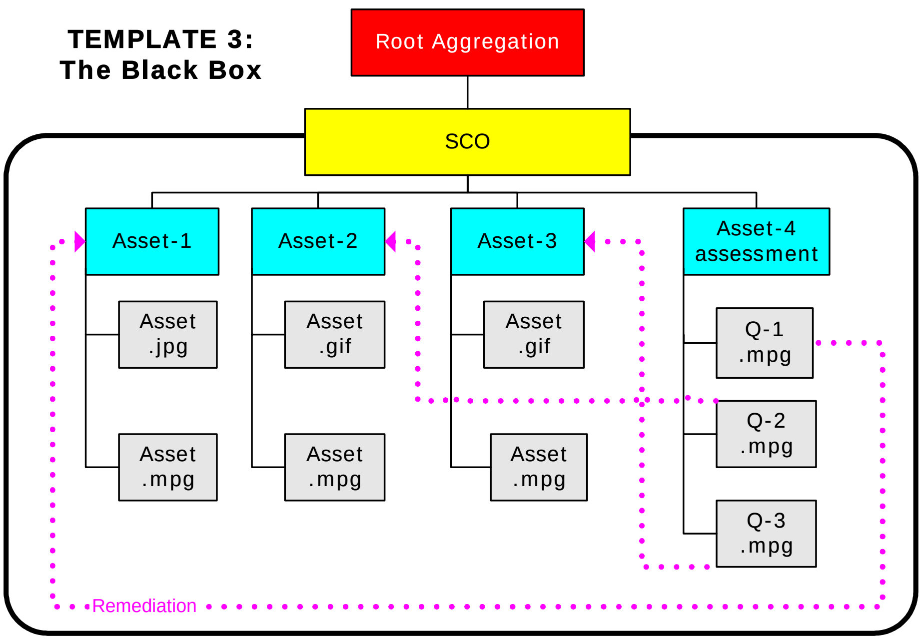
This template represents a SCO composed of multiple "pages" of assets. The SCO in this template might represent a course comprised of several lessons and an assessment. If you have no instructional requirement to track the learner's performance in each of the individual lessons (the assets), then creating your lessons as assets within a single SCO may meet all of your reusability needs. Within this SCO, the presentation of the assets does not impact SCORM in any way.



Template 2 Rules:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must complete the SCO.	Root Aggregation Rollup: If All Satisfied, satisfied.
2. To complete the SCO, the learner must complete the assessment in Asset-4 within the SCO.	*No SCORM function*

7.4.3 Template 3: The Black Box

Template 3 contains *no sequencing*. It is a single SCO with intra-SCO branching. The intra-SCO branching may be as complex or as simple as the designer defines. With this type of intra-SCO branching, the learning management system (LMS) does not know what happens inside the SCO. This means the LMS cannot track or report the learner's progress through the content. While this is an effective way to control the learner's instructional experience, it does not permit the flexibility SCORM seeks to provide.



All remediation occurs as intra-SCO branching, so there is no impact to inter-SCO sequencing.

This template could be viewed as a CBT lesson packaged as a single SCO. None of the behaviors occurring inside the "black box" is tracked by the LMS. To complete the "lesson," the learner must receive a score of 100% on the assessment. The learner is remediated from the missed question to the corresponding asset (if Q-1 is missed, the learner remediates to Asset-1, etc.). The learner is allowed two attempts. If the learner fails attempt two, the learner receives the correct answer and the SCO is marked as passed. Again, this template does not require SCORM sequencing, so these behaviors are not described in the table below.

Template 3 Rules:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must complete the SCO.	Root Aggregation Rollup: If All Satisfied, satisfied.

7.4.4 Template 4: Multiple SCOs with Assets

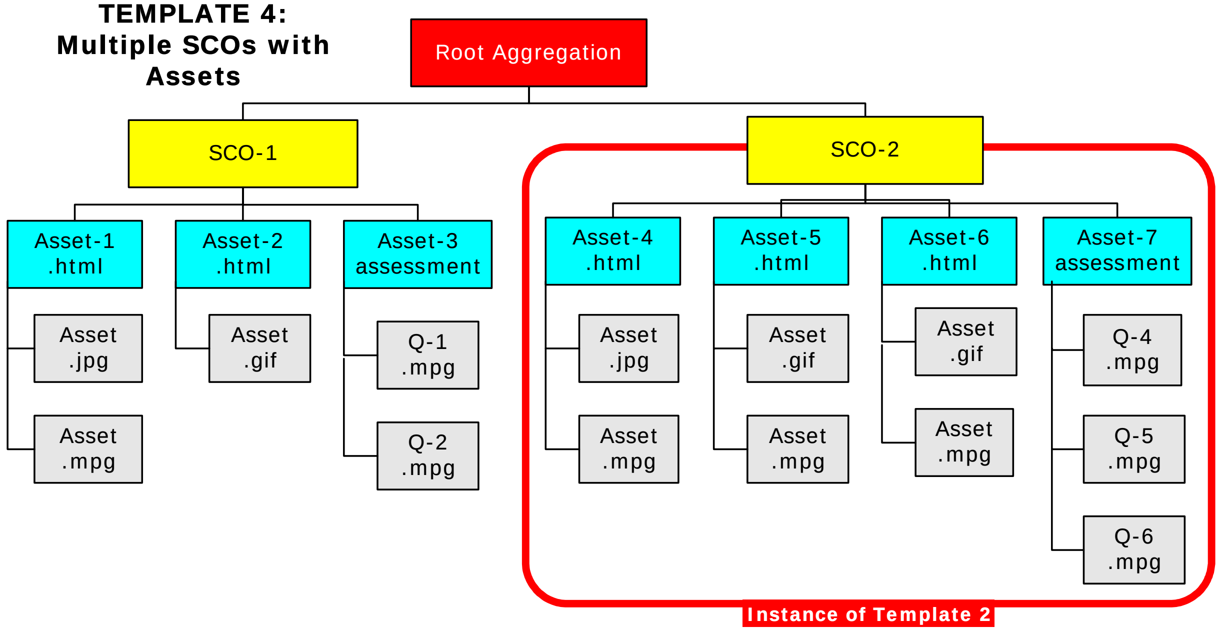
Template 4, or any other template in this guide, can be used in several different ways. Template 4 shows two SCOs in a root aggregation. Here are some of the ways you could interpret the content structure diagram in Template 4:

1) Two assessed learning objectives (the SCOs) in a lesson (the root aggregation)
2) Two assessed segments (the SCOs) in a lesson (the root aggregation)
3) Two assessed lessons (the SCOs) in a module (the root aggregation)
4) Two assessed modules (the SCOs) in a course (the root aggregation)
5) Two assessed lessons (the SCOs) in a course (the root aggregation)
6) Two assessed units (the SCOs) in a course (the root aggregation)

SCO-2 in Template 4 is identical to the SCO in Template 2, showing how these templates can be overlaid to create additional functionality or complexity in a given structure. So, with the ability to “equate” SCORM structures to the traditional instructional design components you are accustomed to working with, and the ability to overlay the templates in this guide, you can essentially create limitless structures of your own.

The rules provided in Application A of Template 4 provide designer-controlled learning while the rules in Application B allow for more learner control of the experience. The set of rules you choose to apply to any template will depend on the learner experience you are trying to create as well as the tracking and training documentation requirements you have.

TEMPLATE 4:
Multiple SCOs with Assets



Template 4 Rules *(Application A)*:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must complete SCO-1 and SCO-2.	Root Aggregation Rollup: If All Satisfied, satisfied.
2. To complete each SCO, the learner must complete the assessments within the SCOs.	*No SCORM function*
3. The learner cannot start SCO-2 until SCO-1 is complete.	SCO-1: Choice=false; Flow=true
4. The learner can return to SCO-1 from SCO-2 at any time.	Root Aggregation: Forward Only=false

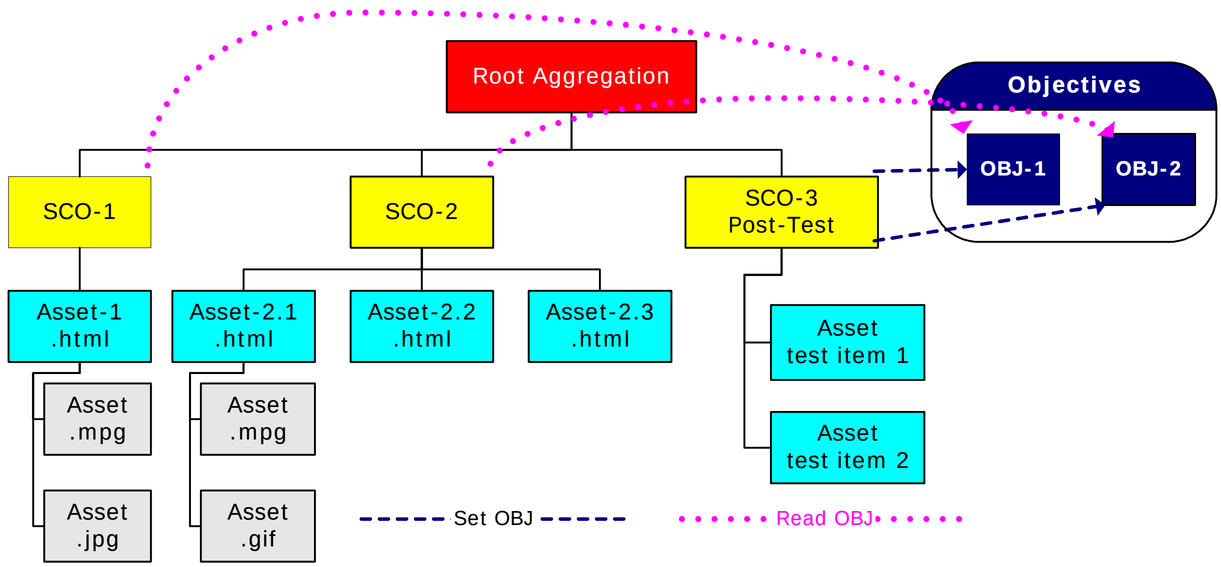
Template 4 Rules *(Application B)*:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must complete SCO-1 and SCO-2.	Root Aggregation Rollup: If All Satisfied, satisfied.
2. To complete each SCO, the learner must complete the assessments within the SCOs.	*No SCORM function*
3. The learner can view the SCOs in any order.	Root Aggregation: Choice=true; Flow=true

7.4.5 Template 5: Remediating Using Objectives

Template 5 presents a sequencing option for learner remediation when you have multiple instructional SCOs. This inter-SCO remediation is tracked by the LMS using objectives (OBJs). The test for this structure exists as a single SCO with two test items (the assets). The post-test (SCO-3) uses objectives to link each test item to its corresponding instructional SCO. Based upon the learner's response to the test item, the OBJ for that item is set to *passed* or *failed*. For failed objectives, the LMS shows the learner the list of corresponding instructional SCOs and the learner can select the SCO to view the remediation.

Suppose the learner fails OBJ-1 and passes OBJ-2. Once the post-test in SCO-3 is complete, the LMS would show the learner the SCOs that **should** be seen again in order for the learner to retake the post-test. In this example, the learner would only see SCO-1 (the SCO corresponding to OBJ-1) listed in the LMS since the learner passed the objective for SCO-2. The learner should then select SCO-1 to complete the remediation and retake the post-test. In the rules, we allowed the learner two attempts to complete this Root Aggregation. Once the learner passes SCO-3, the Root Aggregation is complete. See Template 5 Rules *(Application A)* for specific details.

TEMPLATE 5: Remediating Using Objectives



Template 5 Rules *(Application A)*:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must pass the post-test in SCO-3.	Root Aggregation Rollup: If All Satisfied, satisfied. SCO-1: isRolledup=false SCO-2: isRolledup=false SCO-3: isRolledup=true
2. The learner must complete SCO-1 before attempting SCO-2. The learner must complete SCO-2 before attempting SCO-3.	Root Aggregation: Choice=false; Flow=true
3. To complete SCO-3, both objectives must be passed.	*No unique SCORM function*
4. If the learner fails OBJ-1 in SCO-3, then present SCO-1.	SCO-3: set OBJ-1 SCO-1: skip if OBJ-1 satisfied
5. If the learner fails OBJ-2 in SCO-3, then present SCO-2.	SCO-3: set OBJ-2 SCO-2: skip if OBJ-2 satisfied
6. Allow two attempts for SCO-1, SCO-2, and SCO-3.	SCO-1, SCO-2, SCO-3: Attempt Limit=2
7. If the learner fails SCO-3 on attempt 2, the learner is halted in training and requires manual intervention.	*No unique SCORM function*

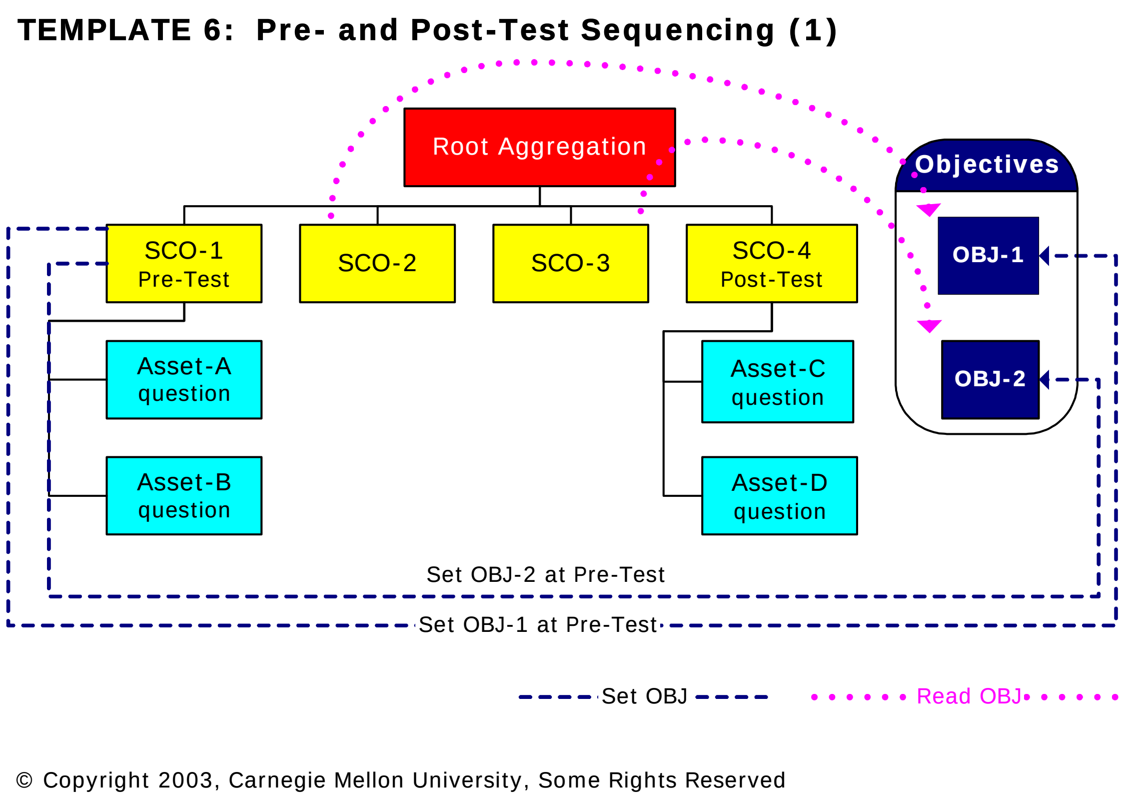
Some templates can be applied in different ways using different behaviors. In Template 5 Rules *(Application B)*, we've given the learner more control over the learning experience. The learner now has the choice to view the content in any order. The learner could even complete the post-test in SCO-3 without first viewing SCOs 1 and 2. The objectives and remediation work the same way in Application B as they do in Application A; however, the learner is now permitted as many attempts as needed to pass the post-test in SCO-3. The table below, Template 5 Rules *(Application B)*, has specific details.

Template 5 Rules *(Application B)*:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must pass the post-test in SCO-3.	Root Aggregation Rollup: If All Satisfied, satisfied. SCO-1: isRolledup=false SCO-2: isRolledup=false SCO-3: isRolledup=true
2. The learner can complete the SCOs in any order.	Root Aggregation: Choice=true; Flow=true
3. If the learner fails OBJ-1 in SCO-3, then present SCO-1.	SCO-3: set OBJ-1 SCO-1: skip if OBJ-1 satisfied
4. If the learner fails OBJ-2 in SCO-3, then present SCO-2.	SCO-3: set OBJ-2 SCO-2: skip if OBJ-2 satisfied
5. Allow as many attempts as needed to complete SCO-3.	*No unique SCORM function*

7.4.6 Template 6: Pre- and Post-Test Sequencing

This template presents a sequencing option for pre- and post-tests of learner knowledge or skills. The pre- and post-tests for this structure exist as individual SCOs. Each post-test item is an individual asset. The testing SCOs are linked to objectives that correspond to test items within the SCO. Based upon the learner's response to the pre-test item, the OBJ is either set to *passed* or *failed*. When the pre-test in SCO-1 is completed, the LMS shows the learner the SCOs corresponding to the missed test questions so the learner can complete the instruction before taking the post-test.

Suppose the learner passes both pre-test items in SCO-1. OBJ-1 and OBJ-2 would be set to *passed*. The learner then has the choice to either skip or complete the instructional SCOs (SCO-2 and SCO-3). However, the learner is required to pass the post-test, so once the pre-test OBJs (OBJ-1 and OBJ-2) are *passed*, the post-test (SCO-4) becomes available to the learner.



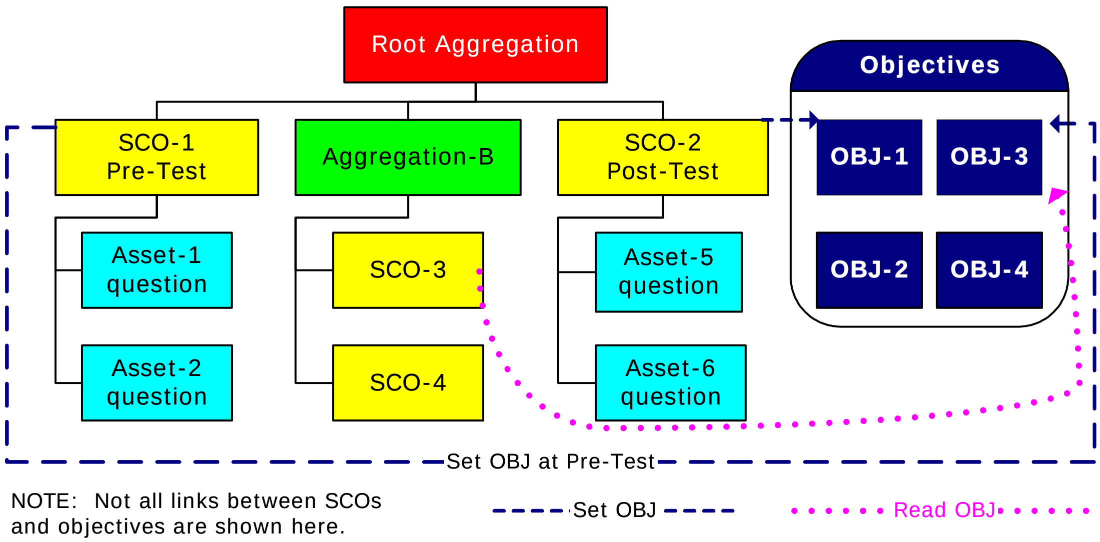
To further expand upon the use of objectives in this template, suppose the learner fails a pre-test item in SCO-1. OBJ-1 (used as a variable) would be set to *failed*, and the LMS would show the learner SCO-1 (the corresponding instruction). Once the learner completed the instructional content in SCO-1, the learner would be able to take the post-test.

Template 6 Rules:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must pass the post-test in SCO-4.	Root Aggregation Rollup: If All Satisfied, satisfied. SCO-1: isRolledup=false SCO-2: isRolledup=false SCO-3: isRolledup=false SCO-4: isRolledup=true
2. The learner must complete the pre-test in SCO-1 before attempting SCO-2 or SCO-3.	Root Aggregation: Choice=false; Flow=true
3. The learner can return to SCO-1 from SCO-2 at any time.	Root Aggregation: Forward Only=false
4. If the learner fails OBJ-1 in SCO-1, then present SCO-2.	SCO-1: set OBJ-1 SCO-2: skip if OBJ-1 satisfied
5. If the learner fails OBJ-2 in SCO-1, then present SCO-3.	SCO-1: set OBJ-2 SCO-3: skip if OBJ-2 satisfied
6. To complete SCO-4, both test items must be passed.	*No unique SCORM function*
7. If the learner fails SCO-4, then the learner is halted in training and requires manual intervention.	*No unique SCORM function*

7.4.7 Template 7: Pre- and Post-Test Sequencing (2)

Template 7 shows a simple way to construct a pre- and post-test "course" (the root aggregation) without remediation. The pre-test sets the objectives (OBJ-3 and OBJ-4) to *passed* or *failed* depending upon the learner's response to the individual test items. If you assume the learner fails OBJ-3 in the pre-test, then the learner would be presented with a list in the LMS showing SCO-3. The learner would select SCO-3 to view the instruction that was not passed in the pre-test. The rules for the diagram require the learner to master the post-test by passing both OBJ-1 and OBJ-2.

TEMPLATE 7: Pre- and Post-Test Sequencing (2)

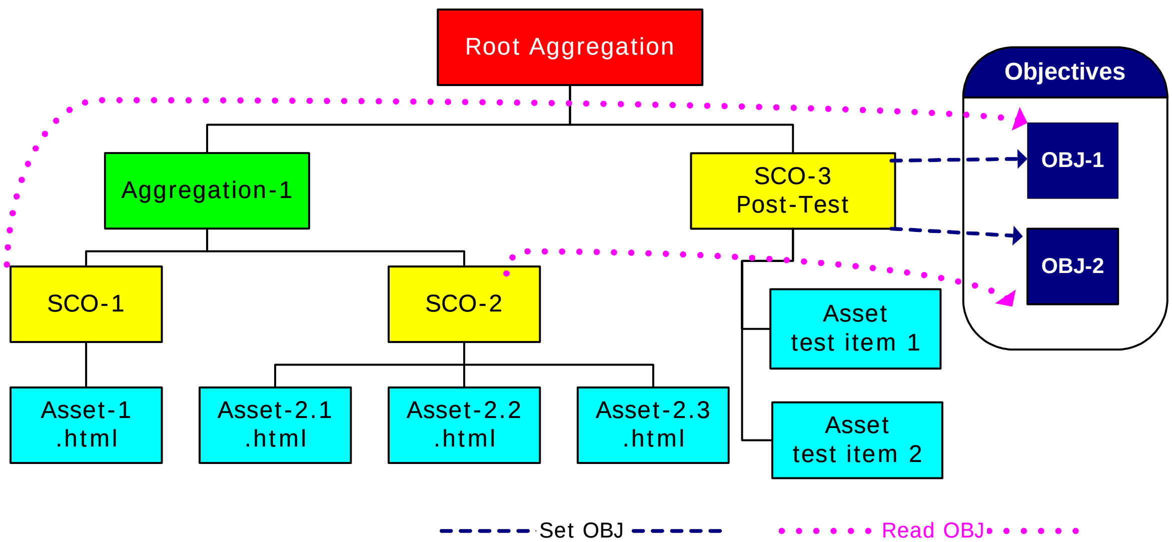


Template 7 Rules:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must pass the post-test in SCO-2.	Root Aggregation Rollup: If All Satisfied, satisfied. SCO-1: isRolledUp=false Aggregation-B: isRolledUp=false SCO-2: isRolledUp=true
2. The learner must complete the pre-test in SCO-1 before attempting Aggregation B or SCO-2.	Root Aggregation: Choice=false; Flow=true
3. The learner can return to SCO-3 from SCO-4 at any time.	Aggregation-B: Choice=true; Flow=true; Forward Only=false
4. If the learner fails OBJ-3 in SCO-1, then present SCO-3.	SCO-1: set OBJ-3 SCO-3: skip if OBJ-3 satisfied
5. If the learner fails OBJ-4 in SCO-1, then present SCO-4.	SCO-1: set OBJ-4 SCO-4: skip if OBJ-4 satisfied
6. The learner cannot return to SCO-1 once Aggregation-B is attempted.	Root Aggregation: Flow=true; Forward-Only=true; Choice=false
7. To complete SCO-2, OBJ-1 and OBJ-2 must be passed.	*No unique SCORM function*
8. If the learner fails OBJ-1 or OBJ-2, then the learner is halted in training and requires manual intervention.	*No unique SCORM function*

7.4.8 Template 8: Remediating Using Objectives (2)

Template 8 allows you to control when the learner can access the post-test. In this template, the learner cannot attempt the post-test in SCO-3 until the instruction in Aggregation-1 is complete. If the learner fails either objective in the post-test, the learner will be remediated to the corresponding instructional materials in Aggregation-1.

TEMPLATE 8: Remediating Using Objectives (2)

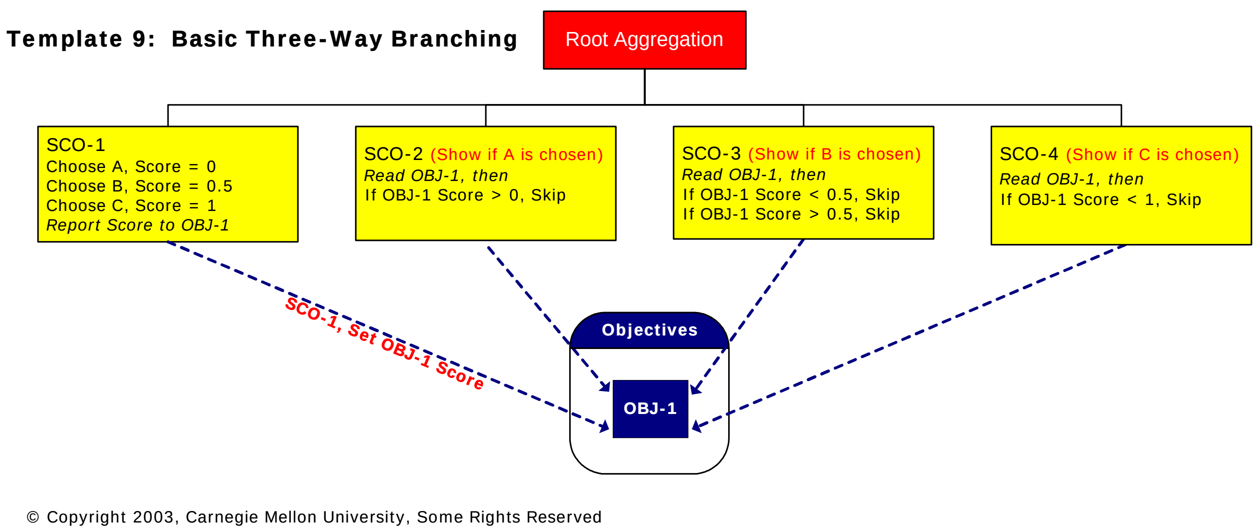


Template 8 Rules:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must pass the post-test in SCO-3.	Root Aggregation Rollup: If All Satisfied, satisfied. Aggregation-1: isRolledup=false SCO-3: isRolledup=true
2. The learner must complete Aggregation-1 before attempting SCO-3.	Root Aggregation: Flow=true; Choice=false
3. The learner can return to SCO-1 from SCO-2 at any time.	Aggregation-1: Forward Only=false
4. To complete SCO-3, both objectives must be passed.	*No unique SCORM function*
5. If the learner fails OBJ-1 in SCO-3, then present SCO-1.	SCO-3: set OBJ-1 SCO-1: skip if OBJ-1 satisfied
6. If the learner fails OBJ-2 in SCO-3, then present SCO-2.	SCO-3: set OBJ-2 SCO-2: skip if OBJ-2 satisfied
7. Allow two attempts for SCO-1, SCO-2, and SCO-3.	SCO-1, SCO-2, SCO-3: Attempt Limit=2
8. If the learner fails SCO-3 on the second attempt, then halt the learner in training and require manual intervention.	*No unique SCORM function*

7.4.9 Template 9: Basic Three-Way Branching

Template 9 shows how you can use simple sequencing rules to accomplish basic adaptive inter-SCO sequencing that is similar to the branching you might have used in traditional CBT lessons. Based upon the learner's choice or decision, represented as a normalized score between –1 and +1, the learner would be directed to another SCO.

Suppose your "course" (the root aggregation) is an adaptive scenario that teaches customer service skills. SCO-1 is the introductory scenario. After reading or viewing the scenario (SCO-1), the learner must make a decision about how to handle the situation with a particular customer. The learner chooses Choice B, which sets the score for SCO-1 to 0.5. Based on the 0.5 score, the learner is directed to SCO-3 for further instruction. This template could be replicated to create as many learner decision points as you desire. For more information on replicating the template, see Model 4. The rules for Template 9 (*Applications A* and *B*) have the same behaviors, but show two alternatives for programming the behaviors.



Template 9 Rules *(Application A)*:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must pass two SCOs (SCO-1 and the *one* other SCO that is chosen by the sequencer). Rule 2 will ensure that SCO-1 is one of the two that is completed.	Root Aggregation: Rollup: if at least two children complete, complete
2. The learner must do SCO-1 first.	Root Aggregation: Flow=true; Forward Only=true
3. Based on the learner's performance on the pre-test, branch to only one of the other three SCOs.	Root Aggregation: Choice=false SCO-1: set OBJ-1 SCO-2: skip if OBJ-1.score > 0 SCO-3: skip if OBJ-1.score < 0.5 or OBJ-1.score > 0.5 SCO-4: skip if OBJ-1.score < 1

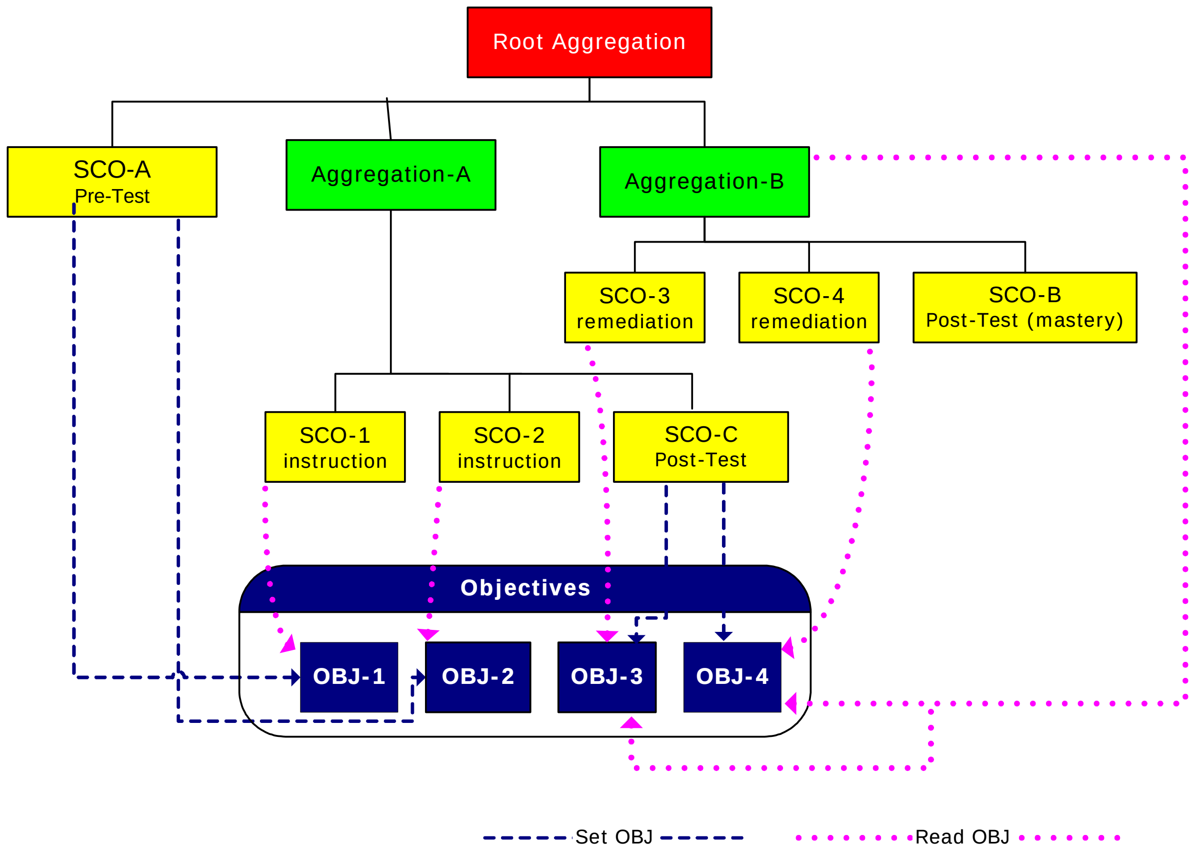
Template 9 Rules *(Application B)*:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must pass two SCOs (SCO-1 and the *one* other SCO that is chosen by the sequencer).	Root Aggregation: Rollup: If at least two complete, complete
2. The learner must do SCO-1 first.	Root Aggregation: Flow=true; Forward Only=true

7.4.10 Template 10: Pre- and Post-Test Sequencing with New Content for Remediation

Template 10 provides a more complex pre- and post-test structure that enables learners to remediate to content that is hidden until needed for remediation. Both the pre- and post-tests are required. Based on the learner's responses to the pre-test in SCO-A, OBJ-1 and OBJ-2 will be set to *passed* or *failed*. Assume the learner fails OBJ-2. A typical LMS will then show SCO-2 on a list. The learner will choose SCO-2 and then take a post-test (SCO-C) to ensure they understand the content from both SCOs 1 and 2. If the learner passes both OBJ-3 and OBJ-4 from SCO-C, then the learner will complete Aggregation-B, thereby completing the root aggregation.

Assume the learner failed OBJ-4 in SCO-C. The LMS will present the learner with SCO-4. SCO-4 contains new instructional material (remediation) that is an enhancement of the content from SCO-2. Since the learner initially struggled with the content, and the learner is required to master the content, the learner must now pass the post-test in SCO-B to complete the root aggregation. If the learner fails SCO-B, then the learner will be halted in training according to these rules. (You could also structure the rules such that the learner passed after a defined number of attempts.) If the learner passes the post-test in SCO-B, then the root aggregation is considered complete.

TEMPLATE 10: Pre- and Post-Test Sequencing With New Content for Remediation



Template 10 Rules:	
Behavior	**SCORM Function**
1. To complete the Root Aggregation, the learner must pass the post-test in SCO-C ***OR*** the post-test in SCO-B.	Root Aggregation: if one child satisfied, satisfied SCO-A: isRolledup=false Aggregation-A: isRolledup=true Aggregation-B: isRolledup=true
2. The learner must complete the pre-test in SCO-A before attempting Aggregation-A. The learner cannot return to the Pre-Test from Aggregation-A.	Root Aggregation: Choice=false; Flow=true; Forward Only=true
3. If the learner fails OBJ-1 in SCO-A, then present SCO-1.	SCO-A: set OBJ-1 SCO-1: skip if OBJ-1 satisfied
4. If the learner fails OBJ-2 in SCO-A, then present SCO-2.	SCO-A: set OBJ-2 SCO-2: skip if OBJ-2 satisfied
5. The learner can return to SCO-1 from SCO-2 at any time.	Root Aggregation: Choice=true; Flow=true; Forward Only=false
6. To complete Aggregation-A, SCO-C must be passed.	SCO-1: isRolledup=false SCO-2: isRolledup=false Aggregation-A Rollup: If All Satisfied, satisfied.
7. The learner will skip Aggregation-B if Aggregation-A is passed.	Aggregation-B: skip if OBJ-1 satisfied and OBJ-2 satisfied
8. If the learner fails OBJ-3 in SCO-C, then present SCO-3.	SCO-C: set OBJ-3 SCO-3: skip if OBJ-3 satisfied
9. If the learner fails OBJ-4 in SCO-C, then present SCO-4.	SCO-C: set OBJ-4 SCO-4: skip if OBJ-4 satisfied
10. If the learner fails SCO-B, then the learner is halted in training and requires manual intervention.	*No unique SCORM function*

7.5 Building Instructional Models Using the Templates

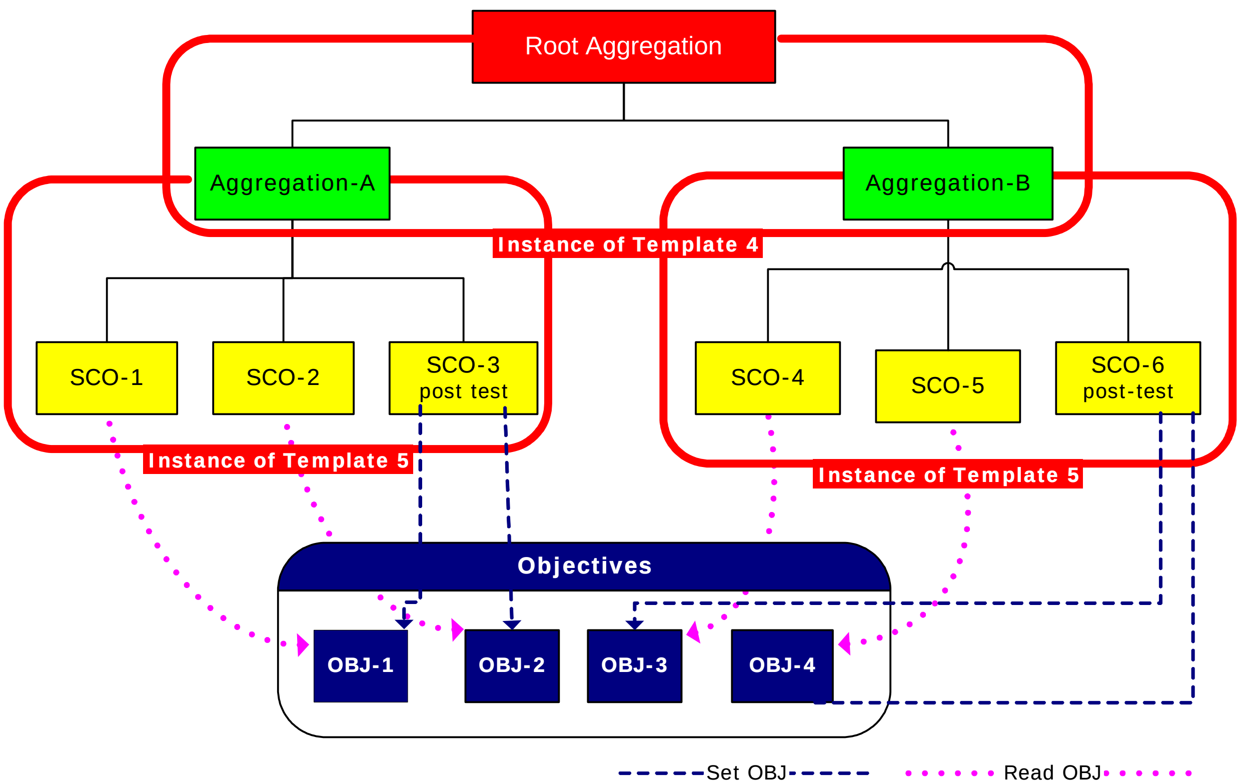
Any template or combination of templates can be "overlaid" on or combined with other templates, creating increasingly complex instructional strategies for courses or lessons. The models that follow show unique combinations of the templates presented in the previous section. The models show the reusability of the templates by labeling each as an instance of a template. In addition, the rules for each model specify from which template, as well as which application of the template, they were obtained. Depending upon how you apply behaviors and rules to the structures, you can achieve a variety of outcomes. These templates and models will provide you with viable sequencing options you can adapt to meet your particular training and educational requirements. For very complex instructional strategies, you can also apply any model or combination of models to another model as was done with the templates.

7.5.1 Model 1: Remediating Multiple Aggregations

Model 1 represents two instances of Template 5 and once instance of Template 4. Template 4 contains two SCOs in a root aggregation. For Model 1, the two SCOs are replaced by Aggregation-A and Aggregation-B that now represent the root aggregation from Template 5. Each aggregation contains three SCOs, one of which is a post-test. The inter-SCO remediation for each aggregation is tracked by the LMS using objectives (OBJs) as global variables.

Each post-test item is linked to an OBJ. Based upon the learner's response to the test item, the OBJ is either set to *passed* or *failed*. In this example, suppose the learner fails a test item in SCO-3. OBJ-1 would be set to *failed* and the LMS would show the learner SCO-1, the SCO that corresponds to OBJ-1. If the learner passes both test items in SCO-3, then the objectives would be set to *passed*, and the learner would proceed to Aggregation-B.

MODEL 1: Remediating Multiple Aggregations



This guide shows two possible applications for Model 1, since each template used to create the model had two possible applications. However, the applications could be combined in any fashion resulting in several more applications for this one model. Suppose you want to create a "course" (the root aggregation) with two units (Aggregation-A and Aggregation-B) each containing two lessons and a post-test (the SCOs). You want the learner to be remediated on a lesson-by-lesson basis, so you create test items tied to their corresponding instruction by objectives. If the learner fails one of the modules, the learner will not be able to complete the course without manual intervention. The rules for Model 1, *Application A* apply.

Model 1 Rules *(Application A)*:		
Behavior	**SCORM Function**	**From Template**
1. To complete the Root Aggregation, the learner must complete Aggregation-A and Aggregation-B.	Root Aggregation Rollup: If All Satisfied, satisfied	4 (A)
2. The learner cannot start Aggregation-B until Aggregation-A is complete.	Aggregation-A: Choice-false; Flow=true	4 (A)
3. To complete Aggregation-A, the learner must pass the post-test in SCO-3.	Aggregation-A Rollup: If All Satisfied, satisfied SCO-1: isRolledUp=false SCO-2: isRolledUp=false SCO-3: isRolledUp=true	5 (A)
4. The learner must complete SCO-1 before attempting SCO-2. The learner must complete SCO-2 before attempting SCO-3.	Aggregation-A: Choice=false; Flow=true	5 (A)
5. To complete SCO-3, both objectives must be passed.	*No unique SCORM function*	5 (A)
6. If the learner fails OBJ-1 in SCO-3, then present SCO-1.	SCO-3: set OBJ-1 SCO-1: skip if OBJ-1 satisfied	5 (A)
7. If the learner fails OBJ-2 in SCO-3, then present SCO-2.	SCO-3: set OBJ-2 SCO-2: skip if OBJ-2 satisfied	5 (A)
8. Allow two attempts for SCO-1, SCO-2, and SCO-3.	SCO-1, SCO-2, SCO-3: Attempt Limit=2	5 A)
9. If the learner fails SCO-3 on attempt 2, the learner is halted in training and requires manual intervention.	*No unique SCORM function*	5 (A)
10. To complete Aggregation-B, the learner must pass the post-test in SCO-6.	Aggregation-B Rollup: If All Satisfied, satisfied SCO-4: isRolledUp=false SCO-5: isRolledUp=false SCO-6: isRolledUp=true	5 (A)
11. The learner must complete SCO-4 before attempting SCO-5. The learner must complete SCO-5 before attempting SCO-6.	Aggregation-B: Choice=false; Flow=true	5 (A)
12. To complete SCO-6, both objectives must be passed.	*No unique SCORM function*	5 (A)
13. If the learner fails OBJ-3 in SCO-6, then present SCO-4.	SCO-6: set OBJ-3 SCO-4: skip if OBJ-3 satisfied	5 (A)
14. If the learner fails OBJ-4 in SCO-6, then present SCO-5.	SCO-6: set OBJ-4 SCO-5: skip if OBJ-4 satisfied	5 (A)
15. Allow two attempts for SCO-4, SCO-5, and SCO-6.	SCO-4, SCO-5, SCO-6: Attempt Limit=2	5 (A)
16. If the learner fails SCO-6 on attempt 2, the learner is halted in training and requires manual intervention.	*No unique SCORM function*	5 (A)

Now, suppose you want to use discovery learning to teach the learner how to start a state-of-the-art dishwasher. You want to slightly restrict the learner's control because the content includes two types of learning. Assume Aggregation-A presents knowledge-based information about the dishwasher and tests the learner's knowledge of the components. Assume Aggregation-B shows two different procedures for starting the dishwasher cycles (SCO-4 and SCO-5). The learner can select the aggregations in any order, since they can start the dishwasher before completing the basic instruction, but the learner has to see both aggregations in order to complete the course.

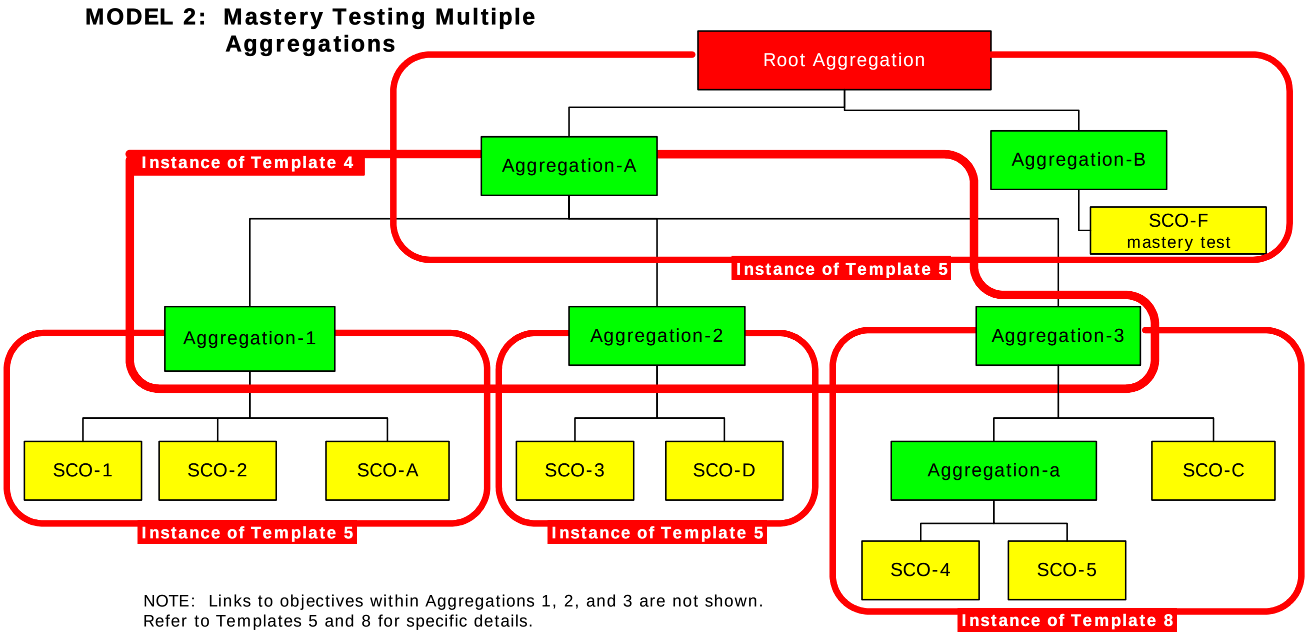
In Aggregation-A, the learner can select the presentation order of the SCOs or take the post-test in Aggregation-A at any time because the order in which the materials are presented is not crucial to understanding the instruction. Since Aggregation-B teaches a procedure, the learner must see the procedures in a predefined order, so SCO-4 is presented before SCO-5 and SCO-5 before SCO-6 (the post-test simulation). The rules for Model 1, *Application B* apply to this example.

Model 1 Rules *(Application B)*:		
Behavior	**SCORM Function**	**From Template**
1. To complete the Root Aggregation, the learner must complete Aggregation-A and Aggregation-B.	Root Aggregation Rollup: If All Satisfied, satisfied	4 (B)
2. To complete each Aggregation, the learner must complete the post-tests within the Aggregations. (See rules 4 and 9).	*No unique SCORM function*	4 (B)
3. The learner can view the Aggregations in any order.	Root Aggregation: Choice=true; Flow=true	4 (B)
4. To complete Aggregation-A, the learner must pass the post-test in SCO-3.	Aggregation-A Rollup: If All Satisfied, satisfied SCO-1: isRolledup=false SCO-2: isRolledup=false SCO-3: isRolledup=true	5 (B)
5. The learner can complete the SCOs in any order.	Aggregation-A: Choice=true; Flow=true	5 (B)
6. If the learner fails OBJ-1 in SCO-3, then present SCO-1.	SCO-3: set OBJ-1 SCO-1: skip if OBJ-1 satisfied	5 (B)
7. If the learner fails OBJ-2 in SCO-3, then present SCO-2.	SCO-3: set OBJ-2 SCO-2: skip if OBJ-2 satisfied	5 (B)
8. Allow as many attempts as needed to complete SCO-3.	*No unique SCORM function*	5 (B)
9. To complete Aggregation-B, the learner must pass the post-test in SCO-6.	Aggregation-B Rollup: If All Satisfied, satisfied SCO-4: isRolledup=false SCO-5: isRolledup=false SCO-6: isRolledup=true	5 (A)
10. The learner must complete SCO-4 before attempting SCO-5. The learner must complete SCO-5 before attempting SCO-6.	Aggregation-B: Choice=false; Flow=true	5 (A)
11. To complete SCO-6, both objectives must be passed.	*No unique SCORM function*	5 (A)
12. If the learner fails OBJ-3 in SCO-6, then present SCO-4.	SCO-6: set OBJ-3 SCO-4: skip if OBJ-3 satisfied	5 (A)
13. If the learner fails OBJ-4 in SCO-6, then present SCO-5.	SCO-6: set OBJ-4 SCO-5: skip if OBJ-4 satisfied	5 (A)

Model 1 Rules *(Application B)*:		
14. Allow two attempts for SCO-4, SCO-5, and SCO-6.	SCO-4, SCO-5, SCO-6: Attempt Limit=2	5 (A)
15. If the learner fails SCO-6 on attempt 2, the learner is halted in training and requires manual intervention.	*No unique SCORM function*	5 (A)

7.5.2 Model 2: Mastery Testing Multiple Aggregations

Model 2 demonstrates how Templates 4 and 5 can be combined into multiple assessed aggregations with a mastery test (SCO-F) for the entire root aggregation. The links to objectives for remediation within Aggregations 1, 2, and 3 are not shown in this model, but they are identical to those shown in Template 5.



Suppose you wanted to create a course (the root aggregation) with several critical lessons (Aggregations 1 – 3). Each lesson builds upon the instruction of the previous lesson, so the lessons must be completed in order. Each of the lessons has several objectives (the SCOs) that are tested and remediated independently. You decide to allow the learner two attempts in each lesson to pass the post-test by providing remediation between the attempts. If the learner successfully passes each of the lessons (thereby completing Aggregation-A), then you allow the learner to attempt the mastery test (SCO-F) in Aggregation-B. If the learner passes the mastery test, then you consider the course complete. However, since each of the lessons are critical, if

the learner cannot pass one of the lessons (Aggregations 1 – 3) after two attempts, you decide they should be automatically halted in training and require manual intervention to proceed. The rules for Model 2 would apply.

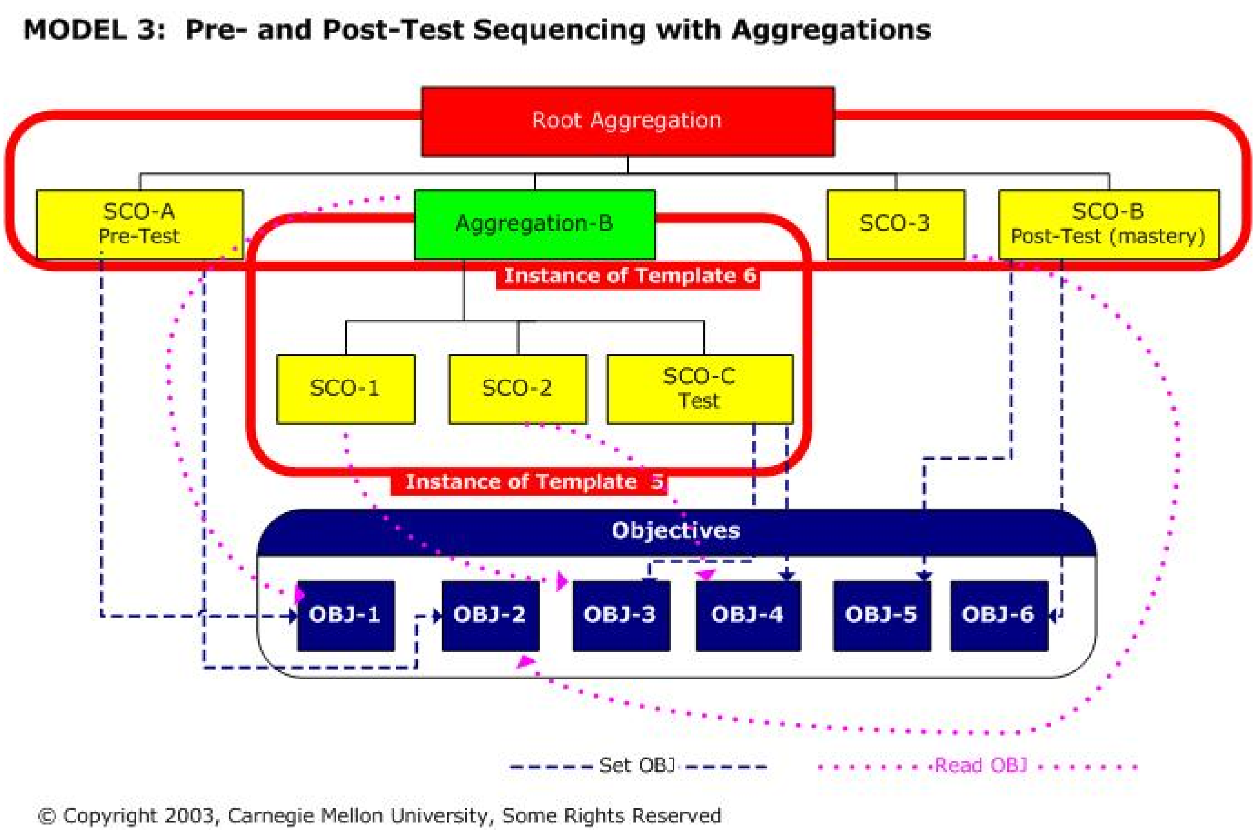
Model 2 Rules:

Behavior	SCORM Function	From Template
1. To complete the Root Aggregation, the learner must pass the mastery test (SCO-F) in Aggregation-B.	Root Aggregation Rollup: If All Satisfied, satisfied Aggregation-A: isRolledup=false Aggregation-B: isRolledup=true	5 (A)
2. The learner must complete Aggregation-A before attempting Aggregation-B.	Root Aggregation: Choice=false; Flow=true	5 (A)
3. To complete Aggregation-A, the learner must complete Aggregation-1, Aggregation-2, and Aggregation-3 in order.	Aggregation A: Choice=false; Flow=true	4 (A)
4. To complete Aggregation-1, the learner must pass the post-test in SCO-A.	Aggregation-1 Rollup: If All Satisfied, satisfied SCO-1: isRolledup=false SCO-2: isRolledup=false SCO-A: isRolledup=true	5 (A)
5. The learner must complete SCO-1 before attempting SCO-2. The learner must complete SCO-2 before attempting SCO-A.	Aggregation-1: Choice=false; Flow=true	5 (A)
6. The learner cannot return to SCO-1 from SCO-2 or SCO-2 from SCO-3 at any time.	Aggregation-1: Forward Only=true	5 (A)
7. If the learner fails OBJ-1 in SCO-A, then present SCO-1.	SCO-A: set OBJ-1 SCO-1: skip if OBJ-1 satisfied	5 (A)
8. If the learner fails OBJ-2 in SCO-A, then present SCO-2.	SCO-A: set OBJ-2 SCO-2: skip if OBJ-2 satisfied	5 (A)
9. Allow two attempts for SCO-1, SCO-2, and SCO-A.	SCO-1, SCO-2, SCO-A: Attempt Limit=2	5 (A)
10. If the learner fails SCO-A on attempt 2, the learner is halted in training and requires manual intervention.	*No unique SCORM function*	5 (A)
11. To complete Aggregation-2, the learner must pass the post-test in SCO-D.	Aggregation-2 Rollup: If All Satisfied, satisfied SCO-3: isRolledup=false SCO-D: isRolledup=true	5 (A)
12. The learner must complete SCO-3 before attempting SCO-D.	Aggregation-2: Choice=false; Flow=true	5 (A)
13. The learner cannot return to SCO-3 once SCO-D is attempted.	Aggregation-2: Forward Only=true	5 (A)
14. If the learner fails OBJ-3 in SCO-D, then present SCO-3.	SCO-D: set OBJ-3 SCO-3: skip if OBJ-3 satisfied	5 (A)
15. Allow two attempts for SCO-3 and SCO-D.	SCO-3, SCO-D: Attempt Limit=2	5 (A)
16. If the learner fails SCO-D on attempt 2, the learner is halted in training and requires manual intervention.	*No unique SCORM function*	5 (A)
17. To complete Aggregation-3, the learner must pass the post-test in SCO-C.	Aggregation-3 Rollup: If All Satisfied, satisfied Aggregation-a: isRolledup=false SCO-C: isRolledup=true	8
18. The learner must complete Aggregation-a before attempting SCO-C.	Aggregation-3: Choice=false; Flow=true	8
19. The learner can view SCO-4 and SCO-5 in any order and can return to SCO-4 from SCO-5 at any time.	Aggregation-a: Choice=true; Flow=false, Forward Only=false	8

Model 2 Rules:		
20. If the learner fails OBJ-4 in SCO-C, then present SCO-4.	SCO-C: set OBJ-4 SCO-4: skip if OBJ-4 satisfied	8
21. If the learner fails OBJ-5 in SCO-C, then present SCO-5.	SCO-C: set OBJ-5 SCO-5: skip if OBJ-5 satisfied	8
22. Allow two attempts for SCO-4, SCO-5, and SCO-C.	SCO-4, SCO-5, SCO-C: Attempt Limit=2	8
23. If the learner fails SCO-C on attempt 2, the learner is halted in training and requires manual intervention.	*No unique SCORM function*	8

7.5.3 Model 3: Pre- and Post-Test Sequencing with Aggregations

Model 3 is a combination of Templates 5 and 6. In this model, a single SCO from Template 6 was replaced with the root aggregation from Template 5. That root aggregation is now Aggregation-B.



Suppose you need to create a course (the root aggregation) that assigns competencies (knowledge, skills, or abilities) to the learner upon successful completion of the instruction. To successfully complete the course, the learner is required to pass a mastery test (SCO-B). You decide that rather than forcing all learners to complete the instructional material you will allow them to take a pre-test (SCO-A). The pre-test will enable learners who already possess the competencies to bypass the instruction and proceed directly to the mastery test (SCO-B). One of the lessons (Aggregation-B) in the course is a critical lesson that teaches an ordered process, so learners who take the lesson are required to complete a post-test (SCO-C) that is a simulation of the process they learned so they can receive credit for passing the competency. For the other lesson (SCO-3), the learner must only view the lesson to receive a competency in that subject area. The rules for Model 3 would apply.

Model 3 Rules:

Behavior	SCORM Function	From Template
1. To complete the Root Aggregation, the learner must pass the post-test in SCO-B.	Root Aggregation Rollup: If All Satisfied, satisfied SCO-A: isRolledup=false Aggregation-B: isRolledup=false SCO-3: isRolledup=false SCO-B: isRolledup=true	6
2. The learner must complete the pre-test in SCO-A before attempting Aggregation-B, SCO-3, SCO-B.	Root Aggregation: Choice=false; Flow=true	6
3. The learner must view all SCOs and aggregations in order.	Root Aggregation: Forward Only=true	6
4. If the learner fails OBJ-1 in SCO-A, then present Aggregation-B.	SCO-A: set OBJ-1 Aggregation-B: skip if OBJ-1 satisfied	6
5. If the learner fails OBJ-2 in SCO-A, then present SCO-3.	SCO-A: set OBJ-2 SCO-3: skip if OBJ-2 satisfied	6
6. To complete Aggregation-B, the learner must pass the post-test in SCO-C.	Aggregation-B Rollup: If All Satisfied, satisfied SCO-1: isRolledup=false SCO-2: isRolledup=false SCO-C: isRolledup=true	5 (A)
7. The learner must complete SCO-1 before attempting SCO-2. The learner must complete SCO-2 before attempting SCO-C.	Aggregation-B: Choice=false; Flow=true	5 (A)
8. To complete SCO-C, both objectives must be passed.	*No unique SCORM function*	5 (A)
9. If the learner fails OBJ-3 in SCO-C, then present SCO-1.	SCO-C: set OBJ-3 SCO-1: skip if OBJ-3 satisfied	5 (A)
10. If the learner fails OBJ-4 in SCO-C, then present SCO-2.	SCO-C: set OBJ-4 SCO-2: skip if OBJ-4 satisfied	5 (A)
11. Allow two attempts for SCO-1, SCO-2, and SCO-C.	SCO-1, SCO-2, SCO-C: Attempt Limit=2	5 (A)
12. If the learner fails SCO-C on attempt 2, the learner is halted in training and requires manual intervention.	*No unique SCORM function*	5 (A)
13. If the learner fails SCO-B, then the learner is halted in training and requires manual intervention.	*No unique SCORM function*	6

7.5.4 Model 4: Traditional CBT Branching with Multiple Decisions

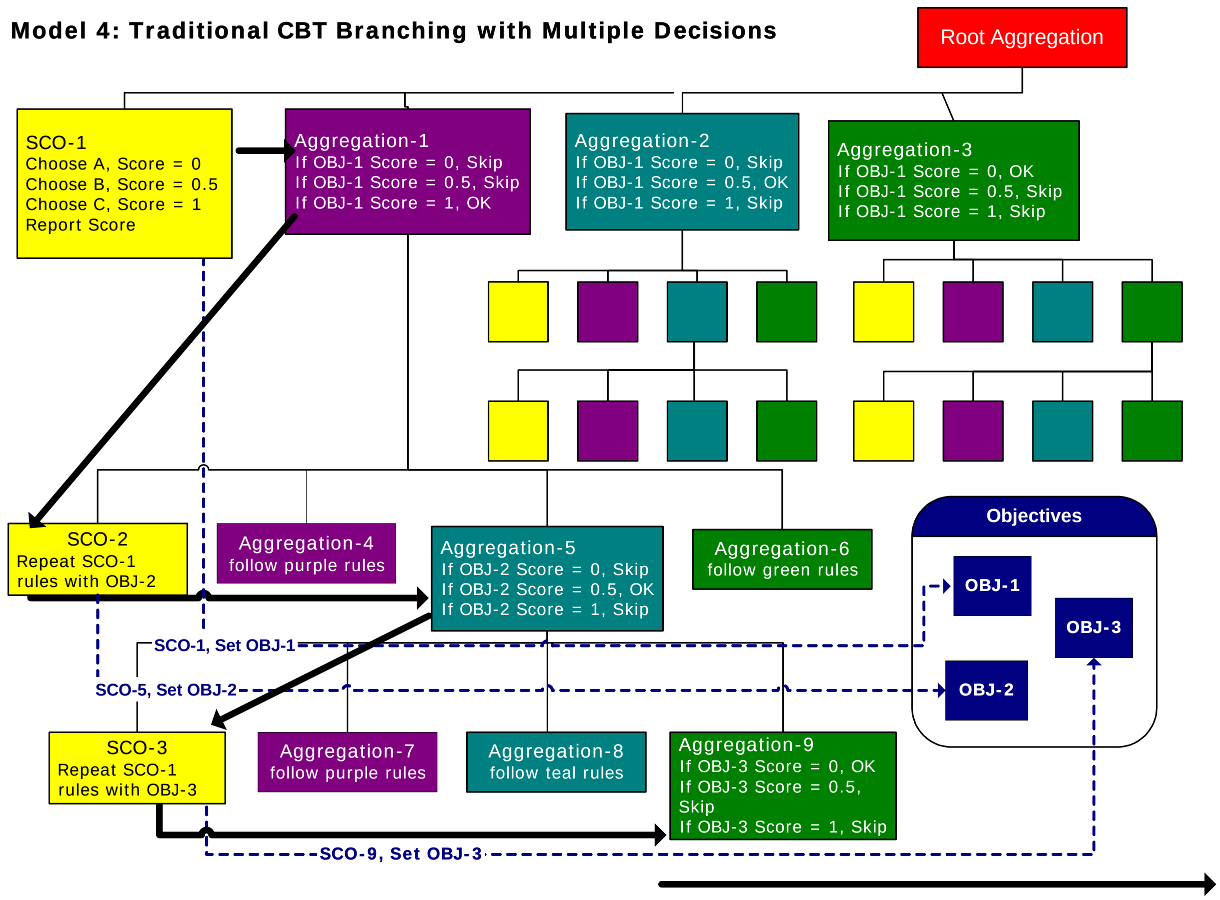
Suppose that your "course" (the root aggregation) teaches customer service skills. You want the learner to have an adaptive scenario-based learning experience. You decide to create a scenario in which the situation changes based on the learner's decision. In this model, SCO-1 presents the learner with a customer situation. After reading or viewing the scenario in SCO-1, the learner decides how to handle the customer.

NOTE: The colors of the aggregations in Model 4 were changed to show the recurrent structure of the rules assigned to each aggregation for three way branching. Each aggregation color represents a unique set of rules that would be programmed for the aggregation.

The learner chooses Choice C (in SCO-1) and then is directed to Aggregation-1. Aggregations can contain rules, but not content, so the rules of Aggregation-1 direct the learner to SCO-2. SCO-2 presents more information about the customer situation. The learner must decide what action to take. If the learner chooses B in SCO-2, then the learner would go to Aggregation-5. The rules on Aggregation-5 send the learner to the next situation with this customer (SCO-3). Assuming the learner selects choice A from SCO-3, the learner would go to Aggregation-9.

In Template 4, Aggregation-9 could be the final scenario you designed, or you could continue building upon the situation with this particular customer through several more decision levels. Aggregations 2 and 3 show other possible routes a learner could take through this instructional material depending upon the decision he made in SCO-1. The objectives in this model are used as global variables. They track the learner's decisions and progress throughout the root aggregation. The rules for Model 4 are specified within the diagram.

Model 4: Traditional CBT Branching with Multiple Decisions



NOTE: In this model, the learner always sees a yellow SCO at each layer in the tree because content can only be contained in a SCO and not in an aggregation.

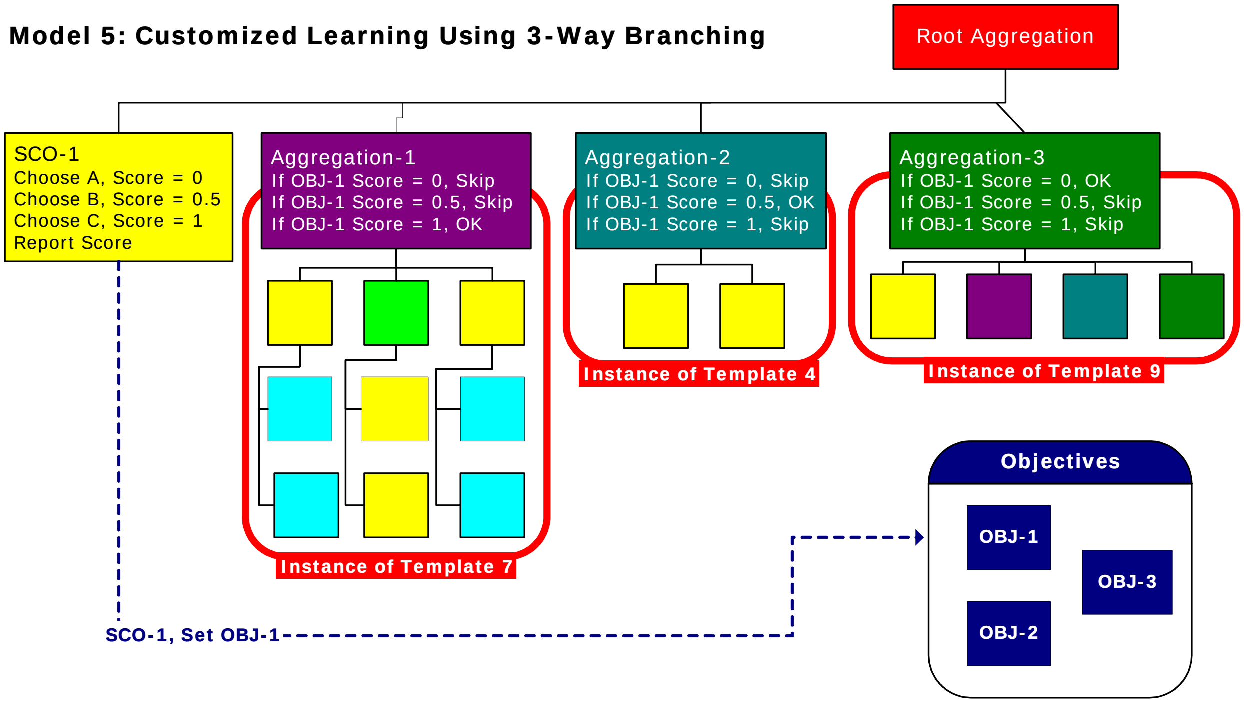
7.5.5 Model 5: Customized Learning Using Three-Way Branching

This model shows how the initial level of Model 4 (or Template 9) can be used to customize learning experiences by presenting learners with a series of choices about who they are, what they do, or what they know. The choices for this type of customization would be predefined learner roles, positions, or competencies.

Using the customer service example presented in Model 4, the learner could select roles such as Customer Service Representative, Customer Service Manager, or Product Specialist. Based on the role the learner selects in SCO-1, Aggregation-1, Aggregation-2, and Aggregation-3 would represent different content structures for the three different positions. Model 5 could remain 3-way branching templates (as show in Model 4) or it could become aggregations of the other templates presented in this guide, thereby giving each role a unique instructional strategy. For Model 5, we've shown the latter using other templates presented in this guide.

Let's assume Aggregation-1, shown in purple, contains the content for a Customer Service Representative. Aggregation-1 is an instance of Template 7. It has a pre-test, content, and a post-test, since customer service representatives may be required to see all activities in a strictly prescribed sequence and must show mastery of the content. Aggregation-2 is an instance of Template 4 that was designed specifically for Customer Service Managers. A Manager might be able to choose the activities she wants to see, since she already has advanced product knowledge. Aggregation-3 was designed for Product Specialists. It reuses the Template 9 for 3-way branching (like in Model 4) to question Product Specialists about their knowledge and to target areas where they need improvement.

Model 5: Customized Learning Using 3-Way Branching



Model 5 can be recreated for customized learning using any of the templates or models in this guide. You can also create your own unique content structure. Simply replace the template's root aggregation with one of the Aggregations shown here.

7.6 Communicating Aggregation Information to the Programmer

Once you have designed your content structure diagram, you will have to explain to the programmer how you want the content structure to be presented to the learner. Since there are currently few tools available to do this, you should consider using an Aggregation Design Specification. This is a single document you can use to communicate the presentation order and sequencing behaviors of each of the children in a particular aggregation. Children may be SCOs or other aggregations within your root aggregation. Each aggregation in a root aggregation requires its own Aggregation Design Specification. An effective Aggregation Design Specification should include the following information that is *specific to the aggregation you are describing*:

1) An objective (if one is required at the aggregation level)
2) Metadata for the aggregation
3) A list of all of the children (may be SCOs or other aggregations)
4) A diagram of the structure (to help the programmer visualize the aggregation)
5) The sequencing behaviors you want for the aggregation

Each Aggregation Design Specification will have the same information in the same location and will be formatted the same way for each aggregation. This makes it easier for the programmers to locate pertinent information during authoring and packaging, thereby making your development process more efficient.

8 Packaging Your Content

Once you have developed all of your physical SCO files, identified the metadata for each SCO and the metadata for the entire content package, and defined your root aggregation, you can prepare to package your content for SCORM. The SCORM content package is a standardized way to exchange digital resources between different learning management systems (LMSs), authoring tools, content repositories, and operating systems.

> **Manifest**
> A manifest is a description of everything contained in your content package. Your programmer will create the manifest as an XML document. The manifest includes your organization and all the metadata for your SCOs.

In traditional instructional design terms, the content package would be everything needed to deliver the course, module, lesson, etc. to the learner. The size of your content package will depend on the structures you've created for your particular content and the manner in which you want them to be delivered to your learners. In SCORM, the content package contains two principal sections:

(1) A manifest that lists all of the resources or assets you want to include in the package, the content structure diagram you created (called the organization), the sequencing rules, and all of the metadata for the SCOs, the aggregations, and the package itself

(2) All of the actual SCO and asset files for the content package

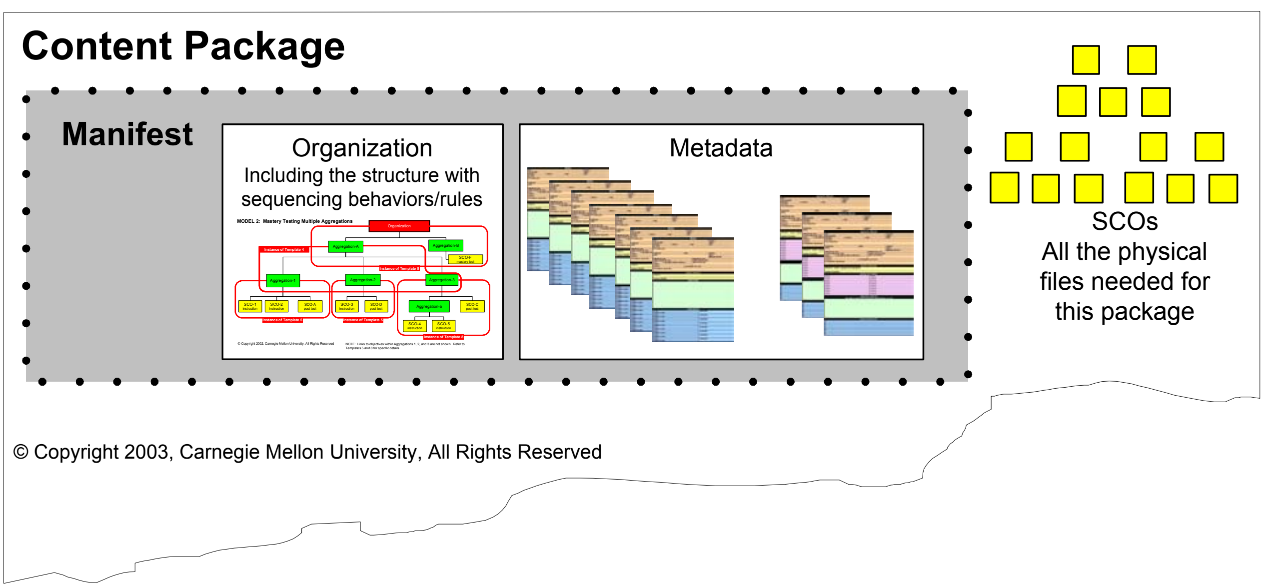


Diagram 8.1: Sections of the content package

Preparing your content package is an excellent time to organize all the files you've used during the development process, including your SCO and aggregation design specifications. Delete or move any incomplete or unused materials, confirm all file names adhere to your naming conventions, and verify that all required metadata fields are complete. Once you've organized all of the files, ensure that the programmer can access them with relative ease. Depending on your process, use either a common file server or a CD-R.

Once the programmer has all of the necessary files, the programmer will create a manifest with your root aggregation and sequencing rules and will store your metadata in the format required for SCORM. Finally, the programmer will create the package with the manifest and all of your SCO content files. Once the package is ready, you can, and should, test the package in any LMS to ensure it functions the way you had intended. If you have access to multiple LMSs, then test your content in as many of them as possible before releasing it.

There are tools that will perform the packaging functions described above. In fact, some authoring tools will not only help you deploy your SCOs, but will also create the entire content package. This guide assumes you are not using a tool that is SCORM 1.3-compliant.

9 Appendix

Sections 9.1 and 9.2 contain the LSAL templates for SCO Design Specifications and Aggregation Design Specifications. These tools, as well as how to use them in your development process, are described in detail in Section 3.5: Using Tools to Design Your SCOs and Section 7.5: Communicating Aggregation Information to the Programmer, respectively.

9.1 LSAL Template for SCO Design Specifications

Metadata		
Title:		Version:
Keyword:		Status:
SCO Description:		Date:
SCO Catalog:		Structure:
SCO Entry:		Format:
Classification Description:		Size:
Purpose:	Classification Keyword:	
Contribute:	Role:	MD Scheme:
Entity:		MD Catalog:
		MD Entry:

Learning Resource Type:		Interactivity Type:
Cost:	Copyright & other restrictions:	Interactivity Level:
Rights Description:		Typical Learning Time:
		Location:

Objective Information
Objective:
Objective Identifier:

Content Outline

Asset Information	
A-1-ID:	Description:
Location in SCO:	
A-2-ID:	Description:
Location in SCO:	
A-3-ID:	Description:
Location in SCO:	
A-4-ID:	Description:
Location in SCO:	
A-5-ID:	Description:
Location in SCO:	
A-6-ID:	Description:
Location in SCO:	
A-7-ID:	Description:
Location in SCO:	
A-8-ID:	Description:
Location in SCO:	

9.2 LSAL Template for Aggregation Design Specifications

Aggregation Metadata		
Title:		**Version:**
Keyword:		**Status:**
Agg Description:		**Date:**
Agg Catalog:		**Structure:**
Agg Entry:		**Format:**
Classification Description:		**Size:**
Purpose:	**Classification Keyword:**	
Contribute:	**Role:**	**MD Scheme:**
Entity:		**MD Catalog:**
		MD Entry:

Learning Resource Type:		**Interactivity Type:**
Cost:	**Copyright & other restrictions:**	**Interactivity Level:**
Rights Description:		**Typical Learning Time:**
		Location:

Objective Information
Objective: (Use only if Aggregation has a unique objective)
Objective Identifier:

Identifiers for Children	
C-1-ID:	**C-9 ID:**
C-2-ID:	**C-10 ID:**
C-3 ID:	**C-11 ID:**
C-4 ID:	**C-12 ID:**
C-5 ID:	**C-13 ID:**
C-6 ID:	**C-14 ID:**
C-7 ID:	**C-15 ID:**
C-8 ID:	**C-16 ID:**

Aggregation Structure (insert diagram of content structure, use children as numbered above (ex: Cx-12 ID)

Sequencing Behaviors
1.
2.
3.
4.
5.
6.
7.
8.

This Page Intentionally Left Blank

10 Glossary

Aggregation
In this document, an aggregation is defined as a parent and its children in a tree structure. Aggregations are used to group related content so that it can be delivered to the learner in the manner you prescribe. Sequencing rules allow you to prescribe the behaviors and functionality of the content within the aggregation as well as how the aggregation relates to other SCOs or aggregations within the same root aggregation.

Application Program Interface (API)
The SCORM API is a standardized method for a sharable content object (SCO) to communicate with the learning management system (LMS) when a learner is interacting with a SCO. There is a specific set of information the SCO can set or retrieve. For example, it can retrieve information, such as a student name, or set values, such as a score.

Asset
Assets are electronic representations of media, text, images, sounds, Web pages, assessment objects, and other pieces of data that can be delivered to a Web client. Assets, like the sharable content objects (SCOs) in which they appear, are highly reusable. In order to be reused, assets are described using metadata so that they are both searchable and discoverable in online content repositories.

Compliance vs. Conformance
The words compliance and conformance, which are often used interchangeably in regular English to convey "accordance", have distinct meanings when used in the context of the adherence of something to a specification or standard. While there is a formal certification program to test for strict conformance of any product to the SCORM guidelines, it is not assumed that the products derived from the processes or procedures in the guide will automatically pass the conformance tests. Thus, this document uses "SCORM-compliant" to refer to the development of materials that comply with the written SCORM guidelines but that are not certified as "SCORM-conformant."

Content Package
The content package is a standardized way to exchange collections of digital resources between different learning management systems (LMSs), authoring tools, content repositories, and operating systems. In traditional instructional design terms, the content package would be everything needed to deliver the course, module, lesson, etc. to the learner. The content package contains two principal entities:
(1) a manifest file that lists all of the resources or assets you want to include in the package, the content structure diagram you created (called the organization), the sequencing rules, and all of the metadata for the SCOs, the aggregations, and the package itself, and (2) all of the actual SCO and asset files for the content package.

Content Structure Diagram
In this document, a content structure diagram is a tree diagram created by the instructional designer for the programmer to show the hierarchy onto which the sequencing rules for the SCOs are applied. The diagram should be followed by a list of the behaviors the instructional designer intends for the learner.

Content Repository
A content repository is a software package designed to manage content in the form of text files, images, etc. throughout its lifecycle, including authoring, versioning control, and distribution of the content. A content repository typically includes the ability to attach metadata to its assets or content and to search for assets or content based on their metadata. A content repository also typically includes security services that allow access to assets by authorized individuals. Content repositories may be distributed to reduce the costs of acquiring large-scale storage devices and to protect data in the event of a disaster.

Learning Management System (LMS)
A Learning Management System (LMS) is a software package used to administer one or more courses to one or more learners. An LMS is typically a web-based system that allows learners to authenticate themselves, register for courses, complete courses and take assessments. The LMS stores the learner's performance records and can provide assessment information to instructors. A learning management system may also perform the following functions: authoring, classroom management, competency management, knowledge management, certification or compliance training, personalization, mentoring, video conferencing, chat, and discussion boards.

Manifest
A manifest is a description of everything contained in your content package. Your programmer will create the manifest as an XML document. The manifest includes your organization and all the metadata for your SCOs.

Metadata
Metadata is "data about data." It is the information that describes what your content is, both the individual pieces (the assets and sharable content objects, or SCOs) and the content packages. Metadata enables instructional designers searching for content or assets to locate them with relative ease and determine whether they will be useful before downloading or requesting rights to your SCOs or assets. See Section 3.4, Locating Your Instructional Materials for more information on selecting and completing metadata fields.

Objective (OBJ)
For SCORM sequencing purposes, an objective is a global variable that allows the learning management system (LMS) to share status values between sharable content objects (SCOs). This gives designers greater flexibility in structuring the content under SCORM guidelines. Depending on the designer's requirements for the instruction, the objective may or may not track actual learner objectives, skills, or abilities.

Organization
The organization is the part of a content package where SCOs are ordered into a tree structure and sequencing behaviors are assigned to them. The organization outlines the entire structure you have created for your content. The organization provides order to the otherwise unordered collection of SCOs and their metadata. Each organization has one top-level aggregation, which we refer to as the root aggregation in this document.

Sequencing

Sequencing is similar to the CBT term "branching" in that it describes and prescribes the manner in which the learner receives the content. In CBT, much of the branching or sequencing occurred within a lesson or course as the learner completed different tasks. However, in SCORM, the learning management system (LMS) sequences all activities between the sharable content objects (SCOs) and the learner, essentially performing all of the sequencing of the content based upon rules created by the designer.

Sharable Content Object (SCO)

A sharable content object (SCO) is a collection of assets that becomes an independent, defined piece of instructional material. SCOs are the smallest logical unit of instruction you can deliver and track via a learning management system (LMS). Refer to Section 2, Defining Sharable Content Objects (SCOs) for more details.

Style Guide

A style guide is the set of established criteria, processes, and procedures a team follows throughout the process of creating instructional materials. It should serve as the handbook or primary reference material for most questions concerning design, layout, and standardization that arise during the content development cycle. Items addressed by the Style Guide may include standardized file naming conventions, required metadata fields, a discussion of the production process, roles of team members, job aids for programs or procedures the team will follow, and the quality control procedures and checklists the team will use. The Style Guide should be minimally impacted by SCORM-specific conventions.

This Page Intentionally Left Blank

11 References

Advanced Distributed Learning Initiative
Sharable Content Object Reference Model (SCORM)
http://www.adlnet.org/

Canadian Core Learning Object Metadata Application Profile; CanCore Learning Object Metadata, Metadata Guidelines, Version 1.1, p. 0-9.
http://www.cancore.org/index.html

IMS Global Learning Consortium, Inc.
Simple Sequencing Specification, Content Packaging Specification, Meta-Data Specification.
http://www.imsproject.org/

International Organization for Standardization
ISO 8601: Data elements and interchange formats -- Information interchange -- Representation of dates and times.
http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=26780&ICS1=1&ICS2=140&ICS3=30

Standard for Learning Object Metadata, IEEE-SA Standard 1484.12.1-2002, IEEE Standards Department, Institute of Electrical and Electronics Engineers, Inc., 2002
http://www.ieee.org/

This Page Intentionally Left Blank

12 Summary of Significant Changes in 1st Edition

The 1st. Edition of this guide incorporates revised and reworded textual explanations that were made to enhance the readability and usability of the guide. In most cases, these changes from previous, preliminary versions were too insignificant to note. However, this summary lists the significant changes included in the SCORM Best Practices Guide for Content Developers (1st Edition) so that users of any previous versions can more easily identify the revisions.

1) The version tracking system for the guide was replaced by an edition system. The First Edition, dated January 30, 2003, supersedes all previous versions of the guide.
2) A description of the usage of compliance versus conformance in this guide was added to the Preface and the Glossary.
3) A new section, Using This Guide, was added to the document before Section 1. This section includes a diagram depicting the processes and products discussed in the guide.
4) Section 2.2: Sharable Content Objects now includes additional material to clarify the roles a SCO can play in your content.
5) Section 3.4: Locating Your Instructional Materials was rewritten. In addition, Table 3.4, Recommended Metadata Fields for SCOs and Aggregations, has changed. Specific changes are noted below.
   a. The title of the second column, "Required" was replaced by "Required by SCORM" to clarify that the metadata field is mandatory for SCORM-compliance.
   b. The following fields were added to the table as SCORM required fields: 1.3, 1.3.1, 1.3.2, 9, 9.1, 9.3, and 9.4.
   c. The "Required by SCORM" value changed for fields 1.6, 2.1, and 2.2.
6) Section 3.5: Using Tools to Design Your SCOs was rewritten to clarify the role of a SCO design specification in the development process.
7) Diagrams 4.2 and 4.3 were simplified and the narrative surrounding them was revised accordingly.
8) Diagram 4.4 was revised to match the additions to the recommended metadata fields.
9) Section 5: Structuring Tests in SCORM was expanded to include more information about the pros and cons of testing with SCORM.
10) Section 6: Navigating in SCORM was expanded to include more information on the navigational issues you may face with SCORM and different LMSs.
11) Section 7: Sequencing Your Content: In all sequencing templates, models, and textual descriptions, the word Organization was changed to Root Aggregation to reflect the language found in the sequencing documents and to more clearly show how the structures can become part of a larger structure. Rules were updated to reflect this change.
12) Section 7.2: The narrative in this section was revised to clarify the roles SCOs can play.
13) Section 7.2.1: Template 1: Rule 2 was deleted.
14) Section 7.2.9: Template 9: All rules were changed to reflect the proper terminology.
15) Section 7.2.10: Template 10: The last rule was modified to correct the rule statement.
16) Section 7.3.3: Model 3: The note at the end of the rules was removed.

17) Section 7.3.4: Model 4: A note was added to the text to clarify the use of different colors in the content structure diagram.
18) Section 8: Packaging Your Content was revised to clarify the components of a content package. Diagram 8.1 was updated accordingly.
19) Section 9.1: The SCO Specification Template was re-titled "LSAL Template for SCO Design Specifications." The metadata section of the specification template was updated to include the mandatory metadata fields.
20) Section 9.2: The Aggregation Specification Template was renamed "LSAL Template for Aggregation Design Specifications." The metadata section of the specification template was updated to include the mandatory metadata fields.